I. ODĂGESCU • I. SMEUREANU • I. ROXIN

VIRUȘI INFORMATICI ȘI BOMBE LOGICE

În decembrie 1987, rețeaua de calculatoare a firmei IBM a suferit un atac al unui „virus", a cărui origine se afla într-o telegramă de felicitare expediată prin rețea de către un utilizator acreditat. Felicitarea făcea să apară pe ecran un pom de Crăciun, dar, în realitate, ascundea un adevărat „cal troian", care – în timpul afișajului – consulta fișierele calculatorului. Apoi, înarmat cu numele persoanelor care au schimbat mesaje cu acest utilizator, trimitea copii ale telegramei (dar și ale programului „Calului Troian") la toți utilizatorii recenzați. Pentru a se debarasa de „virus", IBM a trebuit să închidă sectoare întregi ale rețelei.

ISBN 973-95353-0-5 **Lei 65**

CUPRINS

1. SĂ NE CUNOAȘTEM INAMICII 7

1.1. Ştiri..., informatice 7
1.2. O posibilă definiţie 8
1.3. Virus biologic, virus informatic 11
1.4. Cum s-a născut ideea virusului 12
1.5. Cine sînt atentatorii şi ce vor ei ? 14
1.6. Vaccin biologic şi programe-vaccin 15

2. ANATOMIA ŞI FIZIOLOGIA UNUI CALCULATOR ELECTRONIC 16

2.1. Arhitectura generală a unui calculator 16
2.2. Cine şi cum ne ajută în muncă 22
2.3. Un moment prielnic : încărcarea sistemului de operare MS-DOS 30

3. PUNCTE VULNERABILE ŞI AGENŢI PURTĂTORI 34

3.1. Călcîiul lui Ahile : sistemul de întreruperi . . . 34
3.2. Agenţi purtători : discurile şi dischetele 42
3.3. Formatarea discurilor şi dischetelor 53

4. STRUCTURA ŞI FUNCŢIONALITATEA UNUI PROGRAM DE VIRUSARE 55

4.1. Modulul de implantare 55
4.2. Modulul de propagare şi proliferare a virusului informatic 62
4.3. Modulul distructiv 67
4.4. Modulul de camuflare şi autodistrugere 71

5. INFECTARE CU VIRUŞI ŞI BOMBE LOGICE . . . 75

5.1. Contaminarea prin transfer de fişiere 75
5.2. Infectarea sistemelor izolate 77
5.3. Limbajele de programare a viruşilor 78

6. IDENTIFICAREA ŞI CARACTERIZAREA VIRUŞILOR 80

6.1. Virus latent, virus... productiv 80
6.2. Strategii de căutare 81
6.3. Caracteristici de bază ale virușilor 82

7. TRATAMENT, DEVIRUSARE, IMUNIZARE 88

7.1. Cum să ne păzim de viruşi ? 88
7.2. Detectarea virușilor în sector il de autoîncărcare 89
7.3. Programe de devirusare 90

8. VIRUŞI VECHI, VIRUŞI NOI 103

8.1. Viruşi care infectează fişiere 103
8.2. Viruşi care infectează sectorul de autoîncărcare/ tabela partiţiilor 113

9. PROTECŢIA CONTRA ABUZURILOR... INFORMATICE 118

9.1. Ce este un sistem de protecţie 118
9.2. Mijloace de asigurare a protecţiei 120
9.3. Modele de asigurare a protecţiei 122
9.4. Protecţie, confidenţialitate, siguranţă 124
9.5. Protecţia cu ajutorul... virușilor 128

10. LUPTA CONTINUĂ 133

Bibliografie selectivă 136

1.

SĂ NE CUNOAŞTEM INAMICII

1.1. ŞTIRI... INFORMATICE

• În septembrie 1986, un „intrus" talentat a pătruns în sistemele de calcul din zona San Francisco, sisteme ce includ nouă universităţi, 15 companii din Silicon Valley, nouă companii ale reţelei ARPANET, trei laboratoare guvernamentale. Scopul pătrunderii a fost acela de a deranja cît mai multe calculatoare.

• În decembrie 1987, imaginea unui brad împodobit, originar din Germania, s-a propagat în reţeaua de calculatoare IBM. Datorită înmulţirii rapide a copiilor, circulaţia informaţiilor în reţea a fost serios stînjenită.

• Tot în decembrie 1987, un student a descoperit şi decodificat un program care îşi încorpora copii în programele ce se rulau pe PC. Diverse biblioteci începeau să-şi mărească volumul aparent fără motiv. În unele zile de vineri 13, viteza de lucru a calculatoarelor a fost diminuată, ajungînd ca pe 13 mai 1988 (tot vineri), fişierele să fie complet şterse.

• În aprilie 1988, un virus de 2 kB a infectat un Amstrad, dar a fost detectat înainte de a produce mari pagube.

• La 2 mai 1988, data aniversării lui Mac II, un mesaj de pace a fost afişat pe un număr mare de calculatoare MacIntosh.

• În noiembrie 1988, 6 000 de sisteme din reţeaua ARPANET şi INTERNET au suferit de cea mai mare epidemie de gripă informatică.

• Atenţie la jocuri, care constituie excelente mijloace pirat de contaminare.

• Un potențial șomer a introdus un virus în calculatoare, distrugînd 168 000 înregistrări.

• Excrocherii informatice, miliarde pierdute au condus la întocmirea unor dosare complete de anchetă privitoare la aspectele financiare și juridice ale sinistraților informatici.

• Un excelent dosar în ceea ce privește această formă de delicvență care a atins 250 000 unități PC, stații de lucru și rețele este dat publicității în decembrie 1988. El descrie două virusuri identificate pe Atari și MacIntosh : nVIR și Scores.

• În ianuarie 1989, în revista *Science et Vie*, un autor estimează că problema virușilor nu va avea efecte nefaste asupra evoluției tehnologiei.

• În numărul său din februarie-martie 1989, revista *Science et Vie* descrie diferite vaccinuri contra virușilor : *Vaccine, Virus RX, Interferon, Virus Detective, Kill-Scores*. Precauțiile au impresionat puternic.

1.2. O POSIBILĂ DEFINIȚIE

Știrile anterioare alese dintr-o foarte mare varietate de exemple ilustrează tipurile majore de programe care pot să „atace" alte programe din memoria unui calculator.

Virusul informatic este un mic program „ucigaș" care se află ascuns în memoria calculatorului, așteptînd un mic „semnal" pentru a ieși la iveală. El se autocopiază pe discuri flexibile, stricînd, învălmășind sau ștergînd conținutul altor programe sau date. Ca și cel biologic, virusul informatic preia de la organism (deci, calculator) o celulă vie (programe) și o „păcălește", reproducînd mii de replici perfecte ale virusului original, ceea ce conduce la dezorganizarea informației din sistem. Apoi, „infecția" este răspîndită de la un calculator la altul prin intermediul utilizatorilor care, ori de cîte ori își schimbă dischetele între ei, își vor transmite și „microbul".

Deci, un *virus informatic* este un program care posedă proprietatea de a introduce copii executabile ale lui însuși în alte programe. Fiecare program infectat poate, la rîndul său, să infecteze alte programe. Această definiție nu satisface din punct de vedere științific ; totodată, nu sînt cunoscute lucrări de specialitate care să abordeze în detaliu tema virușilor. Totuși, există anumiți autori care tratează fie și numai enunțiativ asemenea aspecte.

Un program este considerat că reprezintă un virus dacă are proprietățile următoare :

1) modifică programe de utilizator prin includerea propriilor sale structuri în acestea ;

2) modificările provocate nu se referă numai la programe, ci și la grupuri (pachete, sisteme) de programe ;

3) el recunoaște că un program este deja infectat ;

4) dacă recunoaște un program deja modificat, el va interzice o nouă modificare a acestuia ;

5) programul infectat prezintă de acum înainte proprietățile $1 \div 4$.

Dacă programul nu posedă simultan aceste proprietăți el nu poate fi considerat ca un virus, în sensul strict al termenului.

Programele „atacatoare" pot fi de tip : vierme, bacterie, cal troian, virus, bombă logică etc.

Viermele este un program care, odată ajuns într-o rețea de calculatoare, devine activ într-o stație de lucru în care nu se rulează nici un program.

Calul troian este un program care aparent este folositor dar, întocmai strămoșului său, are scopul de a distruge cît mai mult.

Bacteria este un program ce se înmulțește rapid și trăiește în sistemul gazdă ocupînd procesorul și memoria. Ea se fixează în memorie și în calculatorul central, provocînd paralizarea nodului central.

Virusul este un program ce încorporează copii ale sale în cadrul altor programe, infectîndu-le ; la rîndul lor, acestea din urmă pot infecta alte programe, deoarece, de obicei la început — prin intermediul sistemului de operare — se inserează o copie a codului virusului în corpul programului-gazdă. În acest fel crește puțin mărimea fișierului-obiect (fig. 1.1.). Cînd noul program infectat este apelat, virusul pe care îl conține va fi activat și își va îndeplini funcțiile distructive, precum și cele de infectare

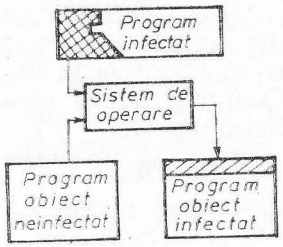


Fig. 1.1. Infectarea unui program

a unor noi programe. Efectul distructiv nu este sesizat imediat, ci se poate manifesta după un anumit timp (ore, zile, luni).

După cum s-a amintit, virusul nu va încerca să infecteze un program deja atacat, deoarece o infecție repetată conduce la o creștere considerabilă a lungimii și implicit la o detectare facilă.

Bomba logică este un cal troian ce atacă fișierele ; un astfel de program poate avea acces la diferite zone ale memoriei, la care utilizatorul nu are acces în mod normal. Spre deosebire de celelalte tipuri de soft agresor, bomba logică se caracterizează printr-un efect distructiv puternic și întotdeauna necontrolat.

Bombele cu întîrziere sînt secvențe de programe introduse în diferite locuri în cadrul unui anumit sistem, la o anumită dată și oră, sau la realizarea unor condiții special impuse apriori, ele acționează distructiv, determinînd ștergerea unor fișiere, blocarea sistemului, reformatarea discului Winchester, distrugerea fișierelor-sistem etc. De menționat faptul că bombele de acest tip pot să apară și cu întîrzieri relative față de sesiunea de lucru. Detectarea unei bombe cu întîrziere este dificilă, motivul constînd în aceea că sistemul poate funcționa corect o bună perioadă de timp pînă la declanșarea bombei.

Un compilator infectat (fig. 1.2) introduce un cal troian în fișierul-obiect al oricărui alt program (de exemplu, în programul de acces în sistem). Ori de cîte ori acest program va fi recompilat, se va insera un fragment de cod ce va permite accesul în sistem pe baza unei parole (cunoscută numai de către proiectantul calului troian).

Fig. 1.2. Infectarea unui fișier obiect

1.3. VIRUS BIOLOGIC, VIRUS INFORMATIC

Noțiunea de virus informatic specifică sistemelor de calculatoare poate fi analizată comparativ cu cea de virus biologic, deoarece ambele noțiuni au foarte multe proprietăți comune. Redăm în continuare o paralelă între virușii amintiți :

Virus biologic	*Virus informatic*
Atacă unele celule ale corpului.	Atacă unele programe din componența unui sistem de calcul.
Transformă informația ereditară din celulă.	Manipulează un program, determinîndu-l să execute sarcini diferite de cele prevăzute.
Celulele infectate produc noi viruși.	Programele infectate generează noi programe infectate.
O celulă infectată nu atacă o altă celulă infectată.	Un program infectat nu atacă alt program infectat.
Un organism atacat poate să nu prezinte nici un simptom timp îndelungat.	Un program infectat poate funcționa mult timp fără erori.
Nu toate celulele intrate în contact cu virusul sînt infectate.	Programele pot fi făcute imune la unii viruși.

Cum este posibil ca programele să se comporte într-un calculator la fel ca virusul într-un organism ? Pentru a răspunde la această întrebare trebuie să ne familiarizăm cu arhitectura unui sistem de calcul, mai precis, cu următoarele noțiuni de bază : hardware, software, microprocesor, unitate centrală, memorie RAM, memorie ROM/EPROM, memorii de masă (dischete, disc dur), periferice (ecran, imprimată), sistem de operare, cod-sursă, cod-obiect, cod executabil, compilator, editor de legături, memorie de lucru etc. Considerînd acest lucru lămurit, credem că paralela între cei doi viruși devine facilă.

1.4. CUM S-A NĂSCUT IDEEA VIRUSULUI

Ideea este la fel de veche ca și calculatorul electronic. Însuși von Neumann, părintele bazelor logice ale calculatorului, a recunoscut încă de la conceperea acestei mașini că există o anumită vulnerabilitate a ei, deoarece folosește aceeași memorie pentru a păstra date și instrucțiuni-program. Un program construit pentru a prelucra date poate, cu unele modificări subtile, să prelucreze chiar și programe.

Pornindu-se din acest punct, unii cercetători americani au produs, în 1959, un joc numit *Core War* (războiul memoriei), care simulează „lupte" între programe diferite din memoria calculatorului. Programele interne, pe care le putem numi „organisme", încearcă să se anihileze unul pe altul, în final rămînînd un singur învingător.

Dar, la începutul erei informatice, calculatoarele erau izolate între ele și deci pericolul unei lupte între programe era inexistent. De asemenea, controlul persoanelor care aduceau lucrări ce trebuiau executate la calculator și al celor ce preluau rezultatele nu ridica probleme deosebite.

Apariția și dezvoltarea unor rețele puternice de calculatoare la nivel național și internațional, conectate prin linii de legătură amplifică și complică cerințele de protecție și confidențialitate a informației. De aceea, un „organism" cu autoreproducere are posibilitatea de a se manifesta. Deci, pe măsura dezvoltării și complicării mijloacelor, metodelor și formelor de automatizare a proceselor prelucrării informației crește vulnerabilitatea ei.

În ansamblul căilor de distrugere, deformare sau scurgere a informației, cele mai mari probleme le ridică acțiunile neautorizate ale personalului centrului de calcul, utilizatorilor sistemului sau ale altor persoane din afara sistemului. Aceste acțiuni constau în folosirea fără aprobare a informației, modificarea ei sau furtul de date chiar de pe liniile de legătură.

Riscul pierderii, distrugerii sau furtului de informații devine una din cele mai alarmante probleme, deoarece

infracțiunile conduc la obținerea unor profituri sau avantaje economice, financiare, militare ori de altă natură pentru indivizi, organizații sau chiar state.

În tabelul 1.1, pentru o perioadă de șapte ani, sînt ilustrate principalele cazuri de infracțiuni. După cum se observă, în ultimii doi ani s-a produs o scădere a numărului infracțiunilor datorită măsurilor de protecție fizică, organizatorică și tehnică a informațiilor.

Tabelul 1.1. INFRACȚIUNI INFORMATICE

Anul	Fraude financiare	Furt de informație	Utilizare neautorizată	Vandalism	Total
1	3	6	2	4	15
2	10	5	10	8	33
3	23	19	6	6	54
4	16	17	18	15	66
5	25	20	11	11	68
6	25	15	12	7	59
7	25	7	4	6	43

Caracteristicile diferitelor infracțiuni sînt : sabotajul, furtul purtătorilor, copierea, copierea neautorizată, înșelarea sistemului de identificare etc.

Prin anii 1983-1984 un realizator de programe (nu un oarecare, ci chiar acela care a elaborat versiunea originală a sistemului Unix) a demonstrat cum poate fi realizat un „organism" cu autoreproducere, numit virus. Mai mult, un alt realizator oferă în schimbul unei sume minime de doi dolari o copie a instrucțiunilor pentru crearea propriului organism. Deci, primul pas pentru răspîndirea în masă a acestui flagel a fost făcut. Să ne mai gîndim la posibilitatea de copiere pe dischete, într-un număr apreciabil, a acestui parazit ? Credem că nu este cazul.

Totuși, se poate observa că astfel de „organisme prietenoase" nu au reușit să pătrundă în sistemele care asigură securitatea informațiilor vitale ale SUA. Acest lucru este o realitate, pentru că măsurile antivirale sînt foarte severe.

În sfera largă a utilizatorilor unele mesaje de avertizare, precum „Nu schimbați discuri !", „Nu copiați programe !" etc., au condus cît de cît la o răspîndire mai restrînsă a virușilor.

Așa cum moda vestimentară este în general trecătoare, poate și moda virușilor informativi va trece... Dar amenințarea va rămîne, iar „binevoitori" se vor găsi tot timpul. Ca urmare, prima măsură împotriva infectării rămîne aceea a neacceptării unei dischete la schimb sau a interconectării în absența măsurilor de siguranță corespunzătoare.

1.5. CINE SÎNT ATENTATORII ȘI CE VOR EI ?

Realizatorii de viruși informatici sînt destul de diferiți. Unii sînt programatori izolați de restul utilizatorilor, care vor să introducă diferite mesaje mai mult sau mai puțin „prietenoase". Unele dintre acestea sînt chiar de natură politică, atenționînd un spectru larg de utilizatori asupra unor opțiuni politice extremiste, fie de stînga, fie de dreapta.

Alți realizatori sînt tineri din școli sau universități, care în teribilismul lor juvenil caută să-și facă simțită prezența prin aceste programe-pirat care atacă.. „la drumul mare" programele pașnice ale calculatoarelor „pacifiste".

Sînt unii furnizori de calculatoare care produc viruși pentru ca, la un moment dat, clienții să apeleze la ei pentru a devirusa, sub diferite forme, programele și calculatoarele infestate. Evident, totul contra cost !

O altă categorie de potențiali creatori sînt tineri care își petrec o mare parte din timpul lor liber în fața ecranului. Motivul ? Vor să atragă atenția lumii asupra performanțelor lor în programare. Un exemplu în acest sens îl constituie tînărul de 23 ani, Drew Davidson, care a creat „virusul păcii".

Categoria cea mai largă de autori de viruși o constituie programatorii care vor să-i pedepsească pe cei ce copiază ilegal software.

Inventivitatea manipulatorilor de viruși este fără limite. Virușii pot fi introduși în programe asemeni agenților „de veghe" ce pot rămîne ani în șir înmagazinați în program. De asemenea, ei pot fi mascați pentru a li se ascunde provenienta și pot fi comandați pentru a se șterge după alte generații de intruși.

1.6. VACCIN BIOLOGIC ȘI PROGRAME VACCIN

Cu foarte puține excepții, pentru fiecare boală există un antidot, un vaccin produs de oamenii în alb (medici, farmaciști, cercetători).

În același timp, mulți întreprinzători dornici de cîștig rapid au vîndut și comercializează în continuare o mulțime de programe destinate protejării calculatoarele. Iată cîteva exemple de astfel de programe, denumite *programe-vaccin* : Virus Rx, Virus-Safe, Vaccinate, Disk-Defender, Antidote, Retro-V, Disk Watcher etc.

Programele-vaccin constituie numai o măsură preventivă în a proteja un disc neinfectat. De obicei, virusul informatic se „ascunde" în puținele locații de memorie ce controlează calculatorul. Vaccinul va identifica și înconjura aceste locații „bolnave". Dacă un alt program va căuta să le utilizeze pe acestea, atunci programul-vaccin va alerta utilizatorul și va opri totul. În această situație trebuie „curățate" memoria și toate discurile, iar programele de bază și discurile vor fi reluate de la sursa originară.

Dar, programele antivirus sînt sigure sută la sută ? Răspunsul este negativ, deoarece există alte programe, numite *retrovirusuri*, care — prin proiectare — apar în sistem la un anumit interval de timp, după ce memoria a fost ștearsă (curățată).

Există unii viruși care infectează partea de interfață cu hard-ul, mărind viteza driverului de disc. Foarte periculoase sînt *programele-vaccin false*, de fapt viruși deghizați ce împrăștie „molima", în loc să o oprească.

2.

ANATOMIA ŞI FIZIOLOGIA UNUI CALCULATOR ELECTRONIC

Ne propunem să prezentăm în acest capitol structura unui calculator electronic şi raporturile dintre diferite „organe" constitutive ale acestuia (anatomia), precum şi funcţiile diferitelor sale structuri „anatomice" (fiziologia). Nivelul de prezentare este subordonat cerinţelor de înţelegere a mecanismului de lucru al virusilor informatici.

Calculatorul electronic numeric este un sistem automat care primeşte, prelucrează şi transmite informaţia codificată sub formă numerică. Un calculator electronic este format dintr-un ansamblu de echipamente *(hardware)* şi de programe *(software)* intercorelate, care asigură introducerea, prelucrarea şi extragerea automată a datelor, potrivit cerinţelor utilizatorilor. Dintre elementele software un rol deosebit revine *sistemului de operare*. El este un sistem de programe de control şi de serviciu care controlează dinamic toate activităţile calculatorului şi asistă utilizatorii în scopul utilizării eficiente a resurselor (memorie, timp unitate centrală, periferice...). Dar care este „anatomia" unui calculator ?

2.1. ARHITECTURA GENERALĂ A UNUI CALCULATOR

Putem privi calculatorul, pentru început, ca o cutie neagră *(black-box)* în care se introduc programele şi datele de prelucrat şi de la care se obţin rezultatele (fig. 2.1.). Dar utilizarea calculatorului „în deplină cu-

Fig. 2.1. Calculatorul-cutie neagră

noştinţă de cauză" presupune cunoaşterea principalelor sale componente.

Pentru a-şi putea exercita funcţiile, calculatorul are în compunerea nucleului său următoarele elemente :

a) *unitatea de memorie internă,* în care se înscriu instrucţiunile programului ce va fi executat şi datele asupra cărora se efectuează operaţiile ;

b) *unitatea aritmetico-logică* (UAL, unitatea de prelucrare a calculatorului, în care ajung datele şi unde se execută operaţiile aritmetice şi logice ;

c) *unitatea de comandă şi control* (UCC), care examinează programul curent, instrucţiune cu instrucţiune, coordonează (supervizează) toate operaţiile efectuate, fiind adevăratul centru de coordonare a sistemului.

Blocul UCC împreună cu UAL formează aşa-numitul *procesor,* iar procesorul alături de memoria internă poartă numele de *unitate centrală de prelucrare* (UCP).

Introducerea programelor şi datelor în memorie se realizează prin intermediul *unităţii de intrare,* iar extragerea rezultatelor din memorie se face prin *unitatea de ieşire* (fig. 2.2). Cu alte cuvinte, schimbul de informaţii cu exteriorul este realizat numai prin mijlocirea memoriei. Întrucît memoria internă are capacitate limitată, înmagazinarea volumelor mari de date se realizează în memorii externe (discuri, dischete, benzi...).

Din punct de vedere structural, elementele unităţii de comandă-control, dispozitivele de comandă, pot fi de *tip convenţional,* în sensul clasic propus de von Neumann, şi *microprogramate,* conform conceptului introdus de Wilkes.

În primul caz, dispozitivul este *cablat* şi o stare corespunde unei *faze de lucru :* într-o fază sînt generate anumite semnale de comandă necesare îndeplinirii unei funcţii (execuţia unei instrucţiuni, a unui ordin etc.).

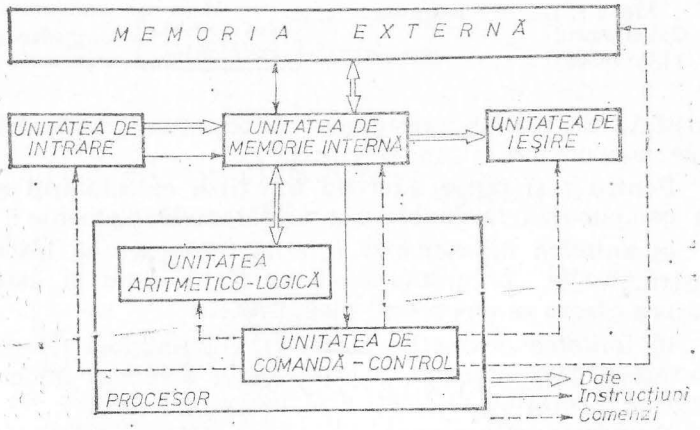


Fig. 2.2. Structura de principiu a unui calculator electronic

În cazul *microprogramat*, unei stări îi corespunde o *microinstrucţiune*, adică o succesiune de biţi ce codifică o serie de comenzi relativ simple. Execuţia unei funcţii mai complexe necesită un *microprogram*. Acest microprogram se află într-o memorie specială, numită *memorie de comandă* (sau *memorie de control*). Unitatea de comandă-control trebuie să fie capabilă să execute un set de instrucţiuni, iar fiecărei instrucţiuni îi corespunde un microprogram înregistrat, începînd cu o anumită adresă, în memoria de comandă.

Simularea unui set de instrucţiuni pe un calculator microprogramat se numeşte *emulare*; microprogramul corespunzător poartă numele de *microprogram de emulare* sau, simplu, *emulator*. Executarea unei instrucţiuni din programul utilizatorului presupune preluarea acestei instrucţiuni din memoria internă, decodificarea ei şi generarea adresei de început a microprogramului de emulare. Toate acestea sînt realizate într-un *bloc emulator* (fig. 2.3).

Urmează derularea microinstrucţiunilor din microprogramul selectat. Fiecare microinstrucţiune ajunge în blocul de decodificare a cîmpurilor microinstrucţiunii şi de generare a comenzilor ce definesc stările procesorului. Trecerea de la o stare la alta a procesorului este sincronizată de către *blocul de generare a impulsurilor de ceas*.

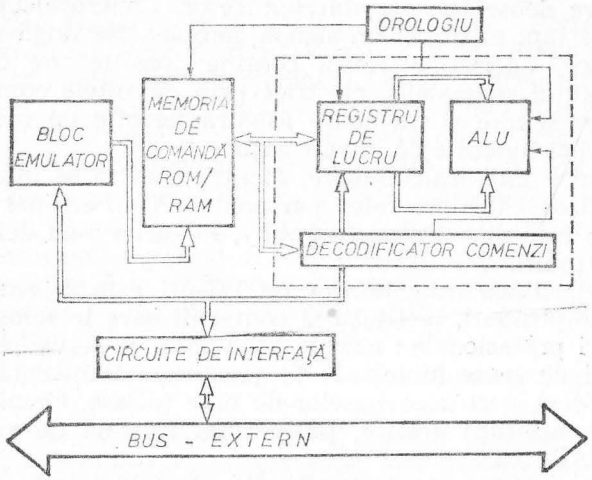


Fig. 2.3. Structura unităţii de comandă microprogramată

O unitate de comandă microprogramată poate fi realizată atît cu circuite discrete (integrate standard), cît şi cu circuite integrate pe scară medie (MSI), largă (LSI) sau foarte largă (VLSI). Creşterea integrării permite realizarea unui bloc funcţional cu puţine cipuri sau chiar a mai multor blocuri funcţionale pe acelaşi cip.

Microprocesoarele sînt circuite care conţin în acelaşi cip ALU, registrele de lucru şi decodificatoarele de comenzi integrate (partea delimitată cu linii întrerupte în fig. 2.3). Dacă setul de instrucţiuni al calculatorului coincide cu setul de instrucţiuni ce poate fi decodificat şi executat de microprocesor, atunci blocul de emulare este inutil şi memoria de comandă se confundă, practic, cu memoria centrală. Sistemele realizate cu microprocesor şi al căror limbaj-maşină coincide cu limbajul microprocesorului se numesc *microcalculatoare*.

Spre deosebire de microprocesor, microcalculatorul este de fapt el însuși un sistem complet. Pe lîngă microprocesor, microcalculatorul conține „pastile" de control al fluxului semnalelor electrice prin circuitele complexe, precum și alte pastile care asigură nevoile de memorie internă și funcțiile de intrare-ieșire.

Dintre microcalculatoare, atenția noastră se îndreaptă spre clasa calculatoarelor personale, PC (Personal Computer). Introduse inițial ca hobby, PC-urile s-au dezvoltat foarte mult.

Majoritatea utilizatorilor de PC-uri este reprezentată de întreprinderi, instituții și companii care le folosesc în scopuri profesionale : calcule economice și statistice, prelucrări de texte (dominate de produsele WordPerfect și WordStar), gestiunea bazelor de date (dBase, Oracle, Paradox), aplicații grafice, proiectarea asistată de calculator (CAD) ș.a.

Firma IBM a proiectat familia de calculatoare PC ca niște sisteme deschise, care se pot dezvolta prin conectarea unor module suplimentare la magistrala universală. Deoarece IBM a publicat datele tehnice referitoare la magistrala folosită, alte firme au putut realiza sisteme *compatibile*, iar IBM-PC a devenit foarte curînd un standard.

Tocmai pe această standardizare se bazează și succesul programelor de virusare, beneficiind de o portabilitate ridicată în gama calculatoarelor PC compatibile. Standardul PC-XT este un calculator IBM echipat cu un microprocesor INTEL 8086 sau 8088. Modelul de pornire, apărut în 1983, era echipat cu o memorie de 64 Ko și un microprocesor cu frecvența ceasului intern de 4,77 MHz. Un PC-AT (Advanced Technology) este echipat cu un microprocesor INTEL 80286 (sau 80386, 80386 SX, 80486) cu frecvența ceasului de 8, 10, 12 sau chiar 33 MHz. Memoria vie este de la 512 Ko în sus. Are un disc dur și un cititor de dischete (de 5"1/4 sau 3"1/2).

În funcție de natura accesului la informația pe care o înmagazinăm, memoria internă a unui PC poate fi cu acces numai în citire și respectiv, cu acces în citire și scriere. *Memoria de tip „citește numai"* (ROM- *Read Only Memory*) se mai numește și memorie permanentă sau memorie moartă, întrucît programele care au fost scrise

în ea sînt fixate o dată pentru totdeauna. Ansamblul programelor încărcate în ROM sau PROM (Programable Read Only Memory — memorie fixă programabilă) — programe cu care este livrat calculatorul de către producător — alcătuiesc așa-numitul *firmware*. Firmware-ul este deci un intermediar între software și hardware.

Memoria cu acces aleator (RAM — *Random Access Memory*), numită și memorie vie, este extrem de flexibilă, informația putînd fi citită și/sau scrisă. Încercînd să simplificăm la maximum lucrurile, putem distinge patru tipuri de memorie vie disponibile pe calculatoarele compatibile PC :

- **memoria convențională** se referă la memoria pe care sistemul de operare MS-DOS o recunoaște în mod curent ; corespunde zonei de memorie de la zero la 640 Ko, zonă în care sînt rulate toate aplicațiile sub MS-DOS ;
- **memoria superioară** *(High Memory)* sau rezervată *(Reserved Memory)* este zona de memorie de la 640 la 1 024 Ko (1024 Ko = 1 Mo). Acest spațiu de 384 Ko, inițial rezervat de IBM unor utilizări diverse, rămîne cel mai adesea nefolosit. Numeroase programe pot utiliza acest spațiu pentru unele drivere de periferice sau pentru a rămîne rezidente ;
- **memoria extinsă** *(Extended Memory)* corespunde întregului spațiu adresabil peste 1 Mo. Pentru un microprocesor 8086 spațiul total adresabil este exact 1 Mo și deci nu putem vorbi de memorie extinsă. Pentru microprocesorul 80 286 spațiul adresabil este de 16 Mo, iar 80386 poate, teoretic, adresa pînă la 4 Go. Avînd în vedere restricțiile de compatibilitate cu 8088 impuse microprocesoarelor 80286 și 80386, memoria extinsă nu poate fi utilizată decît în modul de lucru protejat. Programele care funcționează sub MS-DOS nu pot fi încărcate în memoria extinsă și nici nu pot păstra date în această memorie, întrucît în acest caz microprocesoarele lucrează în „modul real" (folosesc 20 de linii pentru adresare, la fel ca 8088). Prin modul de lucru real se poate adresa, în orice moment, o zonă oarecare a memoriei fizice, în timp ce în modul protejat (microprocesoarele 80286, 80386, 80486) se poate adresa o zonă de memorie virtuală între

16 Mo și 16 Go. În modul de adresare real, din cei 20 biți de adresă primii 16 biți sînt pentru adresa segmentului (fiecare segment are 64 Ko și o adresă multiplu de 16); în modul protejat adresa este dată pe 32 biți: primii 16 biți pentru selecție, iar următorii 16 biți reprezintă deplasarea în cadrul segmentului. Biții de selecție folosesc la identificarea unei intrări într-o tabelă de memorie unde este precizată adresa de bază, pe 24 biți, a segmentului. La aceasta se adună deplasarea și se obține adresa fizică. În zona extinsă pot fi declarate discuri *cache*, discuri virtuale, zone tampon. Recent, mediul de operare Microsoft Windows 3.0 a trecut și la utilizarea memoriei extinse;

- **memoria paginată** *(Expanded Memory)* nu corespunde unei zone de memorie fixate între două adrese (cum au fost primele trei tipuri), dar acoperă parțial sau în totalitate memoria extinsă disponibilă. Pentru adresare se utilizează tehnica paginării (compatibilă cu specificația Lotus/Intel/Microsoft — LIM-EMS), astfel că programele de aplicație lucrînd sub MS-DOS, cu anumite restricții, pot utiliza memoria paginată. Utilizatorul nu are la dispoziție întreaga memorie RAM, o parte a ei fiind ocupată de componentele sistemului de operare, servicii de intrare/ieșire, rutine de întrerupere rezidente și blocuri de control al memoriei.

Modul de adresare a memoriei la nivel fizic, utilizat de programele care se rulează sub MS-DOS, se află și la baza conceperii virușilor informatici. În modul de lucru real întreaga gestiune a spațiului de adrese cade în sarcina programului, în timp ce în modul de lucru protejat, așa cum s-a arătat, sistemul de operare asigură delimitarea spațiului de memorie corespunzător fiecărui task.

2.2. CINE ȘI CUM NE AJUTĂ ÎN MUNCĂ

Folosirea hardware-ului ar fi dificilă și ineficientă fără existența unui sistem de operare (SO). SO este un ansamblu de programe care asigură utilizarea optimă a resurselor fizice și logice ale unui sistem de calcul și con-

cură la pregătirea, punerea în lucru și coordonarea execuției programelor-utilizator. Pentru a-și putea îndeplini rolul arătat, majoritatea SO sînt structurate pe două niveluri:

a) *nivelul logic*, realizat astfel încît să constituie o interfață cu utilizatorul; la acest nivel, utilizatorul comunică cu sistemul de calcul prin comenzi, instrucțiuni și mesaje;

b) *nivelul fizic* este apropiat de hardware și, în general, transparent pentru utilizator. Comunicarea la acest nivel se face prin intermediul unui *sistem de întreruperi* prin care se semnalează anumite evenimente din sistem. La apariția unei întreruperi controlul este dat unor rutine de pe nivelul logic, în scopul tratării întreruperii și al informării utilizatorului despre cauzele care au generat-o. Calculatoarele persoanele actuale au două sisteme de control. La nivelul inferior se află *monitorul ROM*, cunoscut și sub denumirea de *ROM-BIOS*. Al doilea nivel de control este realizat de sistemul de operare MS-DOS (fig. 2.4).

Monitorul ROM-BIOS este un monitor ROM furnizat de firma producătoare de calculatoare ca o parte a hardware-ului. BIOS — acronim de la expresia în limba engleză *B*asic *I*nput/*O*utput *S*ystem (sistemul primar de întreruperi) — este un amestec de soft și hard. Mai mult, constituie veriga absentă care unește cele două „lumi", permițîndu-le să lucreze împreună .Ca orice soft, BIOS este o succesiune de instrucțiuni ce urmează a fi execu-

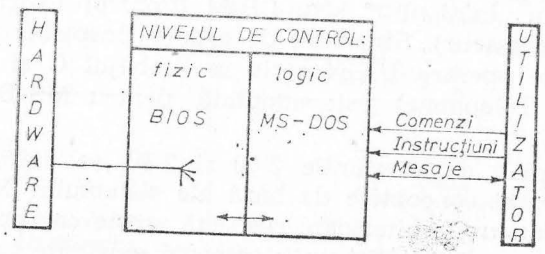


Fig. 2.4. Niveluri de control la un PC

tate de microprocesorul calculatorului. Este memorat în ROM pe placa-mamă și va fi utilizat de SO cu care este dotat calculatorul.

În componența monitorului ROM-BIOS intră :

— programul de autotestare executat la punerea calculatorului sub tensiune (POST — *the power-on self-test program*) ;

— suportul software (drivere fizice) pentru consola monocrom, tastatură, imprimantă, periferice auxiliare și ceas ;

— rutina de startare la inițializarea sistemului ;

— programul care încarcă primul sector din dischetă sau hard disc.

Sarcina esențială a sistemului BIOS este aceea de a asigura independența SO de partea de hardware, astfel încît discurile-sistem pot fi utilizate pe orice mașină de tip IBM-PC.

Pentru un sistem IBM, monitorul ROM-BIOS este scris de IBM fiind protejat (duplicarea lui este interzisă). Pentru calculatoarele IBM compatibile, ROM-BIOS este scris, de regulă, de către un intermediar (firme specializate, cum ar fi American Megatrends, Award Software sau Phoenix Technologies) și vîndut ca parte a hardului.

MS-DOS (*Microsoft Disk Operating System*) este un sistem de operare realizat de firma Microsoft pentru microprocesoarele INTEL 8086, 8088, 80286, 80386 etc.

La 12 august 1981, este lansat pe piață calculatorul IBM-PC cu sistemul de operare Microsoft DOS 1.0 (rebotezat de IBM, PC-DOS). Sistemul păstra încă structura internă a „bunicului" său, CP/M (Control Program for Microprocesseur). Structura de arbore (inspirată din sistemul de operare UNIX scris în limbajul C și realizat de Bell Telephone) este adoptată pentru MS-DOS 2.o în anul 1983.

Începînd cu versiunile 2.00 și 2.10, se definitivează structura și conceptele de bază ale sistemului MS-DOS. Toate versiunile ulterioare respectă aceste concepte, aducînd doar îmbunătățiri unor comenzi existente sau introducînd unele comenzi noi.

MS-DOS este un sistem de operare monoutilizator, iar pentru configurații cu memorie de cel puțin 640 Ko și cu microprocesoarele 80286 sau 80386 este și multitasking.

În anul 1988, a fost lansat MS-DOS versiunea 4.0, iar pentru primul trimestru al anului 1991 este așteptată versiunea 5.0. Noutățile aduse de MS-DOS 5.0 sînt : creșterea memoriei pentru aplicații ; după modelul Windows 3.0 încărcat în mod standard, DOS 5.0 este capabil să execute mai multe taskuri independente, simulînd pe un calculator echipat cu un microprocesor 386 mai multe structuri 8086 dotate fiecare cu propria lor memorie și un management de memorie paginată (multitasking-ul respectă specificațiile DPMI-DOS Protected Mode Interface) ; are un editor de texte în mod ecran care înlocuiește editorul EDLIN ; Quick Basic înlocuiește sistemul GW Basic ; utilitare noi (DOSKEY, UNDELETE, SETVER ș.a.).

Sistemul de operare MS-DOS este mai performant decît monitorul ROM-BIOS și asigură o nouă interfață pentru programul de aplicație (fig. 2.5).

În scrierea programelor de aplicație utilizatorul poate opta pentru folosirea uneia dintre cele trei interfețe : interfața cu MS-DOS ; interfața cu BIOS ; interfața directă cu hardware-ul. Prin folosirea unor niveluri diferite de interfață, programele virus reușesc să-și atingă scopurile fără ca utilizatorul să prindă de veste.

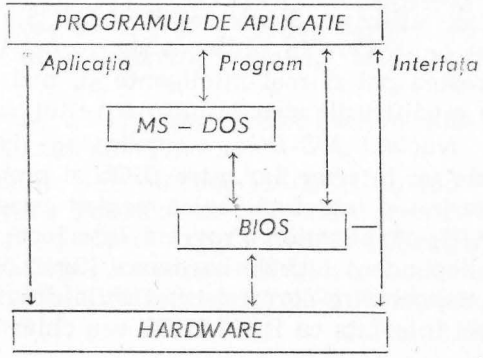


Fig. 2.5. Interfețele programului de aplicație

Sistemul de operare MS-DOS are cinci componente principale: încărcătorul; extensia BIOS; nucleul MS-DOS; interfața utilizator și programele utilitare. Componentele cheie sînt localizate în trei fișiere; extensia BIOS în fișierul IO.SYS (respectiv IBMBIO.SYS pentru IBM); nucleul MS-DOS în MSDOS.SYS (respectiv, IBMBIOSYS), iar interfața utilizator în fișierul COMMAND.COM. Fișierele IO.SYS și MSDOS.SYS sînt fișiere-sistem ascunse (nu pot fi listate prin comanda DIR și nu pot fi copiate sau șterse prin comenzile obișnuite de lucru cu fișierele). Ele trebuie să fie primele fișiere într-un disc utilizat pentru încărcarea sistemului *(disc sistem)*, întrucît ele sînt accesate prin locațiile fizice de pe disc (adresele lor pot fi calculate la încărcarea sistemului de operare pe baza datelor din blocul parametrilor BIOS—BPB).

Fișierul IO.SYS conține trei componente:

— drivere noi care le extind sau le înlocuiesc pe cele din ROM-BIOS;

— rutina de inițializare a acestor drivere;

— un încărcător nou și mai inteligent pentru restul SO, denumit SYSINIT.

Noile drivere pot utiliza componentele BIOS sau extind sistemul ROM-BIOS. Driverele din compunerea sistemului ROM-BIOS sînt încorporate hardware-ului și de aceea pot fi utilizate de rutinele IO.SYS. Memoria și perifericele pot fi testate fără un anumit tip de driver de disc, utilizînd driverele ROM-BIOS. Existența unui alt set de drivere în software reprezintă un avantaj, întrucît acestea pot fi mai inteligente și, mai mult, se adaptează la modificările sistemului (hard-ului) în timp.

Nucleul MS-DOS, cuprins în fișierul MSDOS.SYS, este un intermediar între BIOS și programul de aplicație, oferind o interfață logică pentru programul de aplicație (API—Application Program Interface), care astfel devine independent față de hardware (fig. 2.5). Așa cum s-a mai spus, cei care doresc își pot scrie programele utilizînd direct interfața cu ROM-BIOS sau chiar hardware-ul direct,

dar în acest caz programele vor avea o portabilitate scăzută și nu se vor putea bucura de avantajele oferite de nucleul DOS.

Prin API, programele de aplicație pot avea acces la tastatură, monitor, imprimată sau fișiere-sistem (cum ar fi unitățile logice).

Pe lîngă interfața cu programul de aplicație, nucleul MS-DOS realizează și următoarele funcții:

- *Controlul procesului*, adică gestionarea execuției programelor care presupune încărcarea și inițierea execuției programului, precum și revenirea în SO la terminarea programului.

Cu toate că MS-DOS nu este un SO multitasking, el poate asigura într-o oarecare măsură comunicarea între taskuri utilizînd așa-numitele *conducte* sau *canale de ghidaj (pipes)*. Acestea, idee împrumutată de la UNIX/XENIX, sînt șiruri secvenționale, unidirecționale, de date scrise de un program și citite de alt program. Datele sînt scrise în fișiere temporare pe disc.

Exemplu:

C:\>dir | sort /+25 | more

Comanda anterioară folosește canalul de ghidaj MS-DOS pentru afișarea conținutului directorului în ordine crescătoare, cu pauză după completarea liniilor unui ecran.

- *Gestiunea memoriei* — se referă la alocarea memoriei pentru programele de aplicație. MS-DOS utilizează alocarea memoriei fizice, de aici și limitarea la 640 Ko a spațiului utilizabil de programele care lucrează sub MS-DOS.

Sistemul de operare alocă programelor active memoria sub forma de blocuri de memorie.

- *Gestiunea fișierelor*. Sub nucleul MS-DOS, în mod normal, scrierea și citirea fișierelor se realizează prin intermediul sistemului de gestiune a fișierelor. O aplicație poate avea acces la un fișier în două moduri:

— prin intermediul blocului de control al fișierelor (FCB-File Control Block);

— prin specificarea unui şir ASCII, conţinînd specificaţiile de unitate [u:], numele complet al căii-director ⟨calu⟩ numele *(nume-fis)* şi extensia fişierului ⟨EXT⟩ :

⟨*u* :⟩ ⟨cale⟩ nume-fis ⟨.EXT⟩

MS-DOS va aloca unitatea, directorul şi fişierul creat sau deschis şi îi va asocia o valoare numită identificator logic de fişier (*handle*). Toate accesele relative la fişier (citire, scriere, închidere ş.a.) pot fi executate folosind acest identificator logic. Pentru un astfel de fişier, MS-DOS construieşte un *bloc de parametri* în zona de memorie proprie aplicaţiei care foloseşte fişierului. Dimensiunea acestei zone (deci şi numărul maxim de fişiere care pot fi deschise simultan) depinde de valoarea specificată în comanda de configurare FILES.

Programul *COMMAND. COM* realizează interfaţa utilizatorului cu sistemul MS-DOS şi este constituit din trei părţi : partea rezidentă, partea tranzientă şi o componentă de iniţializare.

Partea rezidentă conţine codul necesar prelucrării comenzii de oprire (caracterele de control CTRL/C şi CTRL/Break), codul pentru erorile de întrerupere, precum şi codul pentru încărcarea părţii tranziente atunci cînd este nevoie.

Partea tranzientă preia comenzile de la tastatură sau din fişierele de comenzi (.BAT) şi conţine comenzile interne. Fiind încărcată în memorie exact înainte de limita celor 640 KB RAM (fig. 2.7), se poate întîmpla ca programul de aplicaţie, dacă este prea mare, să suprascrie partea tranzientă a programului COMMAND.COM.

În acest caz, la terminarea programului partea rezidentă va reîncărca partea tranzientă.

Componenta de iniţializare este încărcată numai la pornirea sistemului şi conţine codul pentru apelarea fişierului AUTOEXEC.BAT. Componenta de iniţializare a programului COMMAND.COM este abandonată după încărcarea fişierului AUTOEXEC.BAT.

Comenzile acceptate de COMMAND.COM pot fi interne, externe şi batch.

Comenzile interne sau intrinseci (exemple : DIR, DEL, COPY, REM etc.) sînt comenzile executate de codul intern al programului COMMAND.COM.

Comenzile externe, sau extrinseci, sînt acele comenzi executate de un cod extern programului COMMAND. COM. Toate programele de aplicaţie (fişierele .EXE şi .COM) sînt încărcate şi executate utilizînd comenzi externe.

Fişierele batch sînt fişiere-text conţinînd o listă de comenzi interne, externe sau numele altor fişiere batch. Interpretorul din partea tranzientă a COMMAND.COM citeşte din fişierul *batch* cîte o linie.

Este important de înţeles mecanismul prin care MS-DOS predă şi preia controlul asupra sistemului. Cînd utilizatorul introduce o comandă, partea tranzientă a COMMAND.COM verifică întîi dacă este vorba de o comandă internă. Dacă nu, utilizînd calea curentă (PATH) caută un fişier de tip .EXE, .COM sau .BAT, cu numele specificat în comandă. Dacă nu este găsit, COMMAND. COM emite mesajul :

Bad command or File name

Cînd este găsit un fişier de tip .EXE sau .COM, utilizînd funcţia EXEC a nucleului MS-DOS, COMMAND. COM încarcă programul în memorie, în zona pentru programe tranzitorii (TPA-*Transient Program Area)* şi îi transferă controlul.

Spre deosebire de sistemul de operare al calculatoarelor *mai-frame* sau al minicalculatoarelor, MS-DOS pierde controlul asupra sistemului atunci cînd este lansat în execuţie un program. Programul aflat în execuţie la un moment dat are un control aproape total asupra resurselor sistemului. Autorii programelor virus vor exploata din plin acest neajuns al sistemului MS-DOS.

Întrucît în memorie pot fi încărcate mai multe programe în acelaşi timp, virusul va coexista cu celelalte programe din memorie, fără a putea fi depistat prea uşor.

Pe de altă parte, cînd execuţia unui program s-a terminat, de regulă, el este înlăturat din memorie şi controlul revine unui alt program. Dar dacă se doreşte, pro-

gramul poate fi construit astfel încît să rămînă în memorie sub forma unui program rezident, TSR (Terminate and Stay Resident); programul este inactiv, dar poate fi activat oricînd la îndeplinirea unor condiții (de exemplu, apăsarea unei taste sau a unei combinații de taste; citirea sau scrierea în disc etc.). Majoritatea virușilor rămîn rezidenți pentru a fi activați cu ușurință. Ei sînt activați chiar de către utilizator sau prin programul acestuia fără măcar să o știe. Cum se poate realiza acest lucru? Ca să putem răspunde la această întrebare, trebuie să înțelegem cum folosește procesorul sistemul de întreruperi.

2.3. UN MOMENT PRIELNIC: ÎNCĂRCAREA SISTEMULUI DE OPERARE MS-DOS

Pentru a înțelege comportamentul virușilor care infectează sectorul de autoîncărcare al discurilor-sistem, prezentăm acum cîteva elemente referitoare la încărcarea sistemului de operare MS-DOS. Încărcarea sistemului este adeseori un moment favorabil pentru pătrunderea virusului în sistem.

Lansarea sistemului începe cu parcurgerea următoarei secvențe sub controlul ROM-BIOS:

1. Este executat programul de autotestare (POST);
2. Se face o anumită inițializare a sistemului;
3. Încărcătorul ROM-BIOS încearcă să citească primul sector al discului sistem (inițial căutat pe unitatea A:).

Prin programul POST se diagnostichează echipamentul (se verifică memoria RAM, perifericele instalate, dacă sînt probleme hard etc.). În faza de inițializare se actualizează tabela întreruperilor, se inițializează unele componente hard, se verifică existența extensiei ROM-BIOS și dacă există este inițializată. Încărcătorul ROM-BIOS este un program scurt care încarcă în memorie primul sector din discul-sistem. Acest program încărcător din ROM-BIOS este același indiferent de sistemul de operare. El asigură parcurgerea primului nivel de încărcare a sistemului.

În funcție de pasul cu care începe încărcarea sistemului, deosebim:

— *încărcarea la rece* (POWER-ON, RESET hard), cînd sînt parcurse toate etapele secvenței;

— *încărcarea la cald* (ALT + CTRL + DEL), cînd se lansează direct încărcătorul ROM-BIOS pentru citirea primului sector al discului sistem.

Urmează al doilea nivel de încărcare a sistemului. Primul sector din discul-sistem conține un încărcător specific sistemului MS-DOS. Încărcătorul MS-DOS — un program scurt și nu foarte rapid — este plasat într-o porțiune a memoriei RAM de către încărcătorul ROM-BIOS. Utilizînd adresele fizice de disc, încărcătorul MS-DOS citește primul fișier din disc (IO.SYS). Este startată o rutină de inițializare care afișează numele producătorului și completează tabela de întreruperi pentru noile drivare definite prin IO.SYS. Odată tabela completată cu adresele corecte, dispozitivele periferice pot fi utilizate. Tabela de întreruperi este situată la baza memoriei RAM (începînd cu adresa 0).

Un al treilea încărcător este lansat pentru încărcarea nucleului sistemului de operare (fig. 2.6). Acest încărcă-

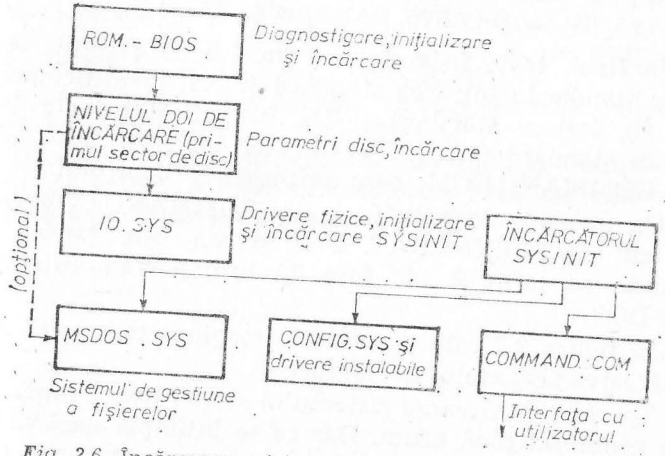


Fig. 2.6. Încărcarea sistemului de operare MS-DOS

tor de nivel trei este reprezentat de programul SYSINIT, parte a fișierului IO.SYS. SYSINIT este un program mai lung și mai inteligent decît primele două încărcătoare, posedînd cunoștințe despre logica discului. Software-ul pentru nucleul MS-DOS este cuprins în fișierul MSDOS. SYS. Încărcarea lui poate fi făcută de către SYSINIT, dar uneori este realizată la nivelul doi de încărcare, depinzînd de constructorul sistemului BIOS.

După terminarea încărcării nucleului de către SYSINIT, controlul este transferat rutinei de inițializare a nucleului pentru actualizarea tabelei de întreruperi, inițializarea driverelor instabile și actualizarea numărului de buffere.

Controlul revine la SYSINIT, care citește fișierul de configurare (CONFIG.SYS). Dacă există, este executat, instalîndu-se și driverele cerute suplimentar. MS-DOS suportă două tipuri de drivere : rezidente și nerezidente sau instalabile.

Driverele rezidente sînt localizate în IO.SYS și, în general, nu pot fi modificate de utilizator. Driverele instabile sînt, de obicei, definite ca fișiere sistem (.SYS) pe disc :

(ANSI.SYS, DISPLAY.SYS, DRIVER.SYS, HIMEM.SYS, PRINTER.SYS, SMARTDRV.SYS, RAMDRIVE.SYS . . .)

În final, toate fișierele sînt închise cu excepția fișierelor standard : intrarea standard (CON, identificator logic Ø), ieșirea standard (PRN, identificator logic 1) și eroare standard (AUX, identificator logic 2). Este încărcat COMMAND.COM, care primește și controlul.

Componenta de inițializare din COMMAND.COM caută fișierul AUTOEXEC.BAT. Dacă există este lansat. Cu aceasta se termină, de fapt, și inițializarea sistemului MS-DOS.

În figura 2.7 este prezentată imaginea memoriei după încărcarea sistemului de operare.

Teoretic, încărcarea sistemului se derulează după cum s-a prezentat pînă acum. Dar ce se întîmplă dacă discul-sistem din care se realizează încărcarea are virusată chiar

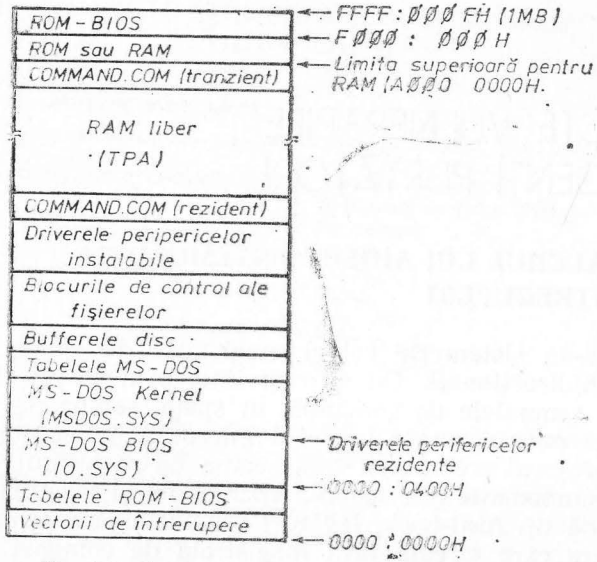


Fig. 2.7. **Geografia memoriei după încărcarea sistemului de operare MS-DOS**

zona de autoîncărcare ? Primul lucru pe care-l va face virusul constă în a se copia pe el însuși în RAM, unde va rămîne rezident. Un virus ajuns în RAM este activ, el poate infecta orice disc ce va fi folosit.

Numai după ce s-a activat, virusul permite sistemului să-și deruleze algoritmul de încărcare.

Dacă încărcarea sistemului solicită mai mult timp decît de obicei, e bine să verificăm dacă nu cumva un virus a valorificat acest moment prielnic.

3.

PUNCTE VULNERABILE ŞI AGENŢI PURTĂTORI

3.1. CĂLCÎIUL LUI AHILE : SISTEMUL DE ÎNTRERUPERI

Într-un sistem de calcul magistrala de comenzi nu este unidirecțională. Cu alte cuvinte, prin ea nu circulă numai semnalele de comandă, în speță, semnalele generate de microprocesor în cazul microcalculatoarelor. Microprocesorul trebuie să țină seama de răspunsul celorlalte componente din sistem. Apar astfel și legăturile inverse de tip feed-back, legături asigurate prin semnalele de stare care circulă prin magistrala de comenzi de la diverse componente spre microprocesor. Dacă aceste semnale de stare n-ar exista, sau dacă n-ar fi luate în considerare, sistemul respectiv ar fi un sistem rigid ce nu s-ar adapta condițiilor schimbătoare ale mediului său înconjurător, ar fi un sistem în care ideea de siguranță ar prima asupra criterilor de performanță.

Întreruperea este tocmai capacitatea sistemului de a suspenda orice activitate aflată în desfășurare, redirecționarea atenției sale spre semnalul generator al întreruperii, rezolvarea cererilor acestui semnal și reîntoarcerea la activitatea suspendată anterior.

Există două mari clase de întreruperi : hardware și software.

Întreruperile hardware sînt inițiate de componentele fizice ale sistemului propriu de pe placa de sistem sau de la canalul de intrare/ieșire.

Pentru un IBM-XT întreruperile hardware sînt prezentate în tabelul 3.1. Pentru un AT lista este ceva mai mare, dar oricum sînt destul de puține și sînt predestinate pentru un sistem hardware dat. Întreruperile hardware nu sînt corelate cu activitățile software și sînt complet transparente pentru programator.

Tabelul 3.1. ÎNTRERUPERILE HARDWARE PENTRU IBM XT

Numărul întreruperii	Utilizare
Întrerupere nemascabilă	Paritate memorie
0	Sistemul de timp (ceas)
1	Tastatură
2	Rețea
3	COM_2
4	COM_1
5	Disc dur
6	Floppy disc
7	Imprimanta

• *Întreruperile software* sînt întreruperi a căror acțiune este programabilă și desfășurată sub control software ; ele nu sînt, cum s-ar putea crede, întreruperi cauzate de software. Întreruperile software pot avea trei categorii de cauze : hardware (de exemplu, întreruperile de ceas) ; de excepție (împărțirea prin zero, căderea tensiunii etc.) ; software (întreruperea este activată printr-o instrucțiune a programului de aplicație). Prin întreruperile software într-un program se asigură controlul interfețelor atît la nivel BIOS, cît și la nivelul nucleului DOS. Autorii de viruși se întreabă : Dacă întreruperile sînt programabile, atunci nu sînt și alterabile (se pot modifica) ? Din fericire, sau din păcate, sînt. Dar să vedem cum se derulează o întrerupere software. Într-un sistem PC-compatibil, la întîlnirea unei întreruperi soft, este inițiat următorul ciclu :

1. Se întrerupe derularea în secvență a programului, executîndu-se automat un salt la o locație de memorie prestabilită în funcție de tipul întreruperii și începe procesul de tratare a întreruperii, utilizînd o rutină de întrerupere specializată ;

2. Rutina de întrerupere încarcă în stivă conținutul curent al registrelor și adresa de revenire, apoi tratează întreruperea ;

3. La terminarea rutinei de întrerupere este restaurat conținutul registrelor și se revine în programul întrerupt, pentru a se executa următoarea instrucțiune.

Totul este relativ simplu. Dar de unde știe procesorul unde găsește rutina pentru tratarea unei întreruperi ? Soluția oferită pentru calculatoarele IBM-PC/XT/AT și PS/2 este foarte elegantă. La baza memoriei, începînd cu locația 0, se află o tabelă a vectorilor de întrerupere. Fiecare intrare în această tabelă este reprezentată de o adresă, adresa de început pentru o rutină de întrerupere. Această adresă este reprezentată pe 4 octeți : 2 pentru segment și 2 pentru deplasarea în cadrul segmentului.

Fiecărei întreruperi îi sînt asociate un număr (tabelul 3.2) și, corespunzător, o intrare în tabela vectorilor de

Tabelul 3.2. ÎNTRERUPERILE BIOS ȘI FUNCȚIILE ASOCIATE LOR

Număr întrerupere (hexa)	Funcție
00	Împărțire prin zero
01	Pas cu pas
02	Întrerupere nemascabilă (NMI)
03	Punct de întrerupere
04	Depășire
05	Imprimare ecran
06—07	(Rezervate)
08	Întrerupere ceas
09	Tastatură
0A	(Rezervată)
0B—0C	Comunicații
0D	Harddisc
0E	Dischetă
0F	Imprimantă
10	Video
11	Verificare echipament
12	Determinarea mărimii memoriei

(continuare tabelul 3.2.)

Număr întrerupere (hexa)	Funcție
13	Dischetă/disc
14	Comunicații
15	Casetă
16	Servicii tastatură
17	Imprimantă
18	BASIC ROM
19	Pornire sistem (la rece)
1A	Setarea timpului
1B	Oprire de la tastatură
1C	Incrementare ceas utilizator
1D	Parametrii video (inițializare)
1E	Parametrii dischetă
1F	Caracteristici videografice
20—3F	Rezervate DOS-ului
40	Redirectare BIOS dischetă
41	Parametrii harddisc
42—45	(Rezervate)
46	Parametrii harddisc
47—49	(Rezervate)
4A	Alarmă utilizator
4B—5F	(Rezervate)
60—67	Rezervate întreruperilor din programele utilizator
68—6F	(Rezervate)
70	Întrerupere pentru ceasul de tip real
71—74	(Rezervate)
75	Redirectare spre NMI
76—7F	(Rezervate)
80—Fo	Rezervate pentru BASIC
F1—FF	Rezervate întreruperilor din programele utilizator

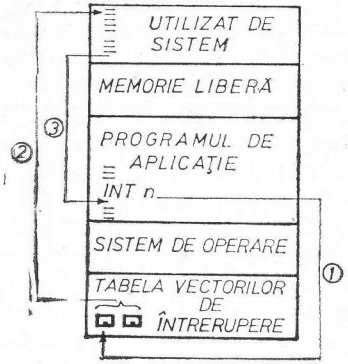


Fig. 3.1. Derularea unei întreruperi software

întrerupere. Prin înmulțirea numărului întreruperii cu 4, procesorul găsește punctul de intrare în tabela de unde va prelua adresa de început a rutinei de tratare. Programul trebuie să furnizeze doar numărul întreruperii, ceilalți pași ai ciclului fiind parcurși automat de procesor (fig. 3.1.).

Există două clase de întreruperi software. Prima clasă, întreruperile BIOS (tabelul 3.2.), cuprinde întreruperile pentru consolă, imprimantă și monitor, întreruperi a căror rutină de tratare se găsește în BIOS (zona ROM). A doua clasă o reprezintă întreruperile MS-DOS (tabelul 3.3.), întreruperi pentru hard disc și dischete. Rutinele de tratare a întreruperilor MS-DOS se găsesc pe discul-sistem.

Tabelul 3.3. ÎNTRERUPERILE MS-DOS

Numărul (în hexa)	Funcția
20	Terminare program
21	Execuția unei funcții DOS
22	Adresa de terminare
23	Adresa pentru tratare caracter oprire (CTRL/C)
24	Adresa pentru tratarea proprie a erorii
25	Citire absolută disc
26	Scriere absolută disc
27	Termină și rămîi rezident (TSR)
28—2E	(Rezervate)
2F	Multiplă (comenzile PRINT, ASSIGN, SHARE, numai după MS—DOS 30)
30—3F	(Rezervate)

Folosirea interfeței BIOS asigură programelor de aplicație performanțe sub aspectul timpului de lucru, dar programele devin dependente de hardware și nu pot fi lansate cînd se lucrează în rețea sau sub medii de operare cum ar fi Microsoft Windows.

Interfața MS-DOS, deși mai lentă, oferă următoarele avantaje :

— independența programelor față de hardware (serviciile MS-DOS vor fi onorate pe orice hardware care suportă sistemul MS-DOS) ;

— suport pentru mediile grafice (cum ar fi Microsoft Windows) ;

— multitasking și lucrul în rețea ;

— servicii extinse pentru gestiunea fișierelor.

Rutinele utilizate de MS-DOS pentru tratarea operațiilor de intrare/ieșire pot fi apelate în orice program de aplicație, facilitînd munca de programare și mărind șansele compatibilității programului cu viitoarele versiuni MS-DOS.

Întreruperea software 21H (INT 21H) este serviciul pentru apelul funcțiilor MS-DOS. Întrucît virușii utilizează frecvent funcțiile MS-DOS disponibile prin INT 21H, în tabelul 3.4 prezentăm o parte din acestea. Întreruperile software cerute prin program nu sînt aleatoare. Ele sînt cerute în mod premeditat, la momente de timp bine definite prin programul care se rulează. Spre exemplu, dacă programul are nevoie de data și ora sistemului, va transmite o întrerupere (08h) și aceste elemente vor fi furnizate în registrele de lucru. Nu este vorba aici de nici un apel („CALL") la o altă rutină, cum se procedează la alte sisteme, și nici un alt program de bibliotecă nu va fi încărcat. Întregul sistem este sub controlul programatorului, oferindu-i astfel posibilitatea să folosească „puterea" acestuia prin programul de aplicație. Astfel, în programele DOS întreruperile joacă un rol deosebit, întregul sistem fiind, de fapt, condus prin întreruperi. Utilizarea rutinelor de întrerupere oferă cel puțin 3 mari avantaje :

Tabelul 3.4. FUNCȚII MS-DOS PRIN ÎNTRERUPEREA 21H

Numărul funcției (hexa)	Funcție
00	Terminarea programului
01	Citirea cu ecou de la tastatură
02	Afișarea unui caracter
03/04	Intrare/ieșire auxiliară
05	Tipărirea unui caracter
06	Intrare/ieșire directă fără ecou de la consolă
09	Afișarea unui șir
0D	Resetarea discului
13	Ștergerea unui fișier, utilizînd FCB
17	Redenumirea unui fișier, utilizînd FCB
19	Obținerea unității curente implicite
1B	Obținerea informațiilor FAT pentru unitatea centrală
21	Citirea în acces direct, utilizînd FCB
22	Scrierea în acces direct, utilizînd FCB
23	Obținerea dimensiunii unui fișier, utilizînd FCB
25	Poziționarea vectorului de întrerupere
29	Analiza unui nume de fișier
31	Păstrarea procesului
36	Obținerea spațiului liber pe disc
3B	Schimbarea directorului (CHDIR)
41	Ștergerea unui fișier
44	Control intrare/ieșire pentru dispozitive (IOCTL)
48	Alocarea memoriei
49	Eliberarea memoriei alocate
4A	Modificarea alocării memoriei (SET BLOCK)
56	Schimbarea intrării în director
57	Obținerea/poziționarea datei și orei unui fișier
5A	Crearea unui fișier temporar unic
5B	Crearea unui fișier nou
5E	Servicii de rețea
62	Obținerea adresei PSP-ului

— *flexibilitate :* toate rutinele de întrerupere pot fi modificate prin simpla modificare a adreselor din tabela vectorilor de întrerupere (încărcare a unor adrese noi corespunzătoare unor prelucrări dorite) ;

— *independența programelor de aplicație față de hardware :* sistemul de întreruperi asigură o interfață logică între program și sistem. Această interfață, ușor de utilizat, este standardizată și definită de IBM, ceea ce asigură portabilitatea programelor pe toate sistemele PC-compatibile (adesea non-DOS) ;

— *modularitate :* prin gruparea mai multor funcții comune sub o întrerupere comună. O singură întrerupere poate astfel trata oricare din funcțiile din gruparea făcută (de exemplu, întreruperea 10H pentru serviciile video are mai multe subfuncții).

Dar de aceste avantaje vor beneficia și programatorii rău intenționați. Am văzut că foarte multe din programele virus rămîn rezidente fără ca utilizatorul sistemului să știe. Virusul va intercepta anumite întreruperi, de regulă cele mai uzitate, înlocuind în tabela vectorilor de întreruperi (fig. 3.2)adresa de început a rutinei de tratare cu propria sa adresă de început. Reușește astfel să preia controlul asupra sistemului. Virusul își poate

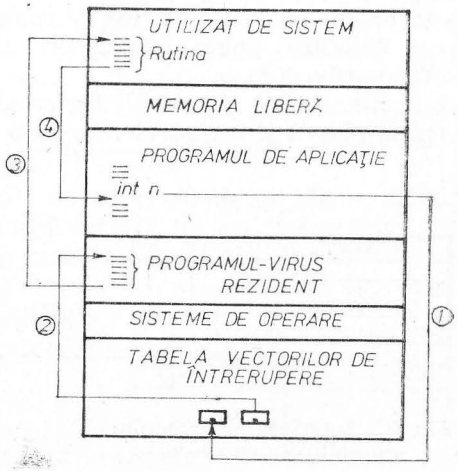


Fig. 3.2. Derularea unei întreruperi interceptate de un virus rezident

desfășura, în taină, activitatea malefică, după care va efectua saltul la adevărata rutină de tratare a întreruperii, lucrurile decurgînd pentru utilizator aparent normal.

3.2. AGENȚI PURTĂTORI : DISCURILE ȘI DISCHETELE

În lucrul cu discurile și dischetele interfața programului de aplicație poate fi realizată la diverse niveluri (fig. 3.3).

La nivel fizic, discul este văzut ca o mulțime de elemente de lungime fixă : volume, cilindri, piste și sectoare. Atît pentru dischete, cît și pentru discul dur întreaga colecție de date formează așa-numitul volum. Datele sînt fizic memorate în volum sub forma unor cercuri concentrice numite piste. Fiecare pistă, la rîndul ei, este împărțită în mai multe sectoare. Fiecare sector are o lungime fixă, de regulă 512 octeți (fig. 3.4).

Discurile flexibile pot fi înregistrate pe o singură față sau pe amîndouă, iar discurile hard conțin mai multe platane pe un suport și pot avea deci un număr variabil de fețe. Toate sectoarele de pe toate fețele care au aceeași distanță față de centru formează un cilindru (mulțimea pistelor cu același număr).

Cilindrul poate fi imaginat ca o stivă verticală de sectoare (fig. 3.4).

Elementele *structurii logice* sînt, ca și la structura fizică, pistele și sectoarele.

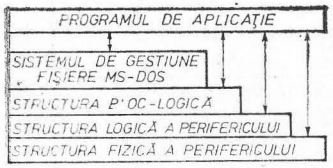


Fig. 3.3. Interfața programului de aplicație cu discul

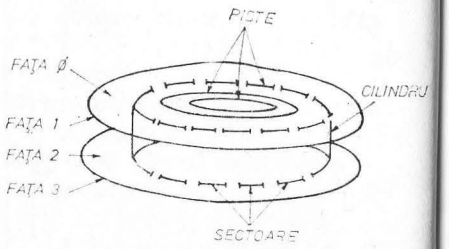


Fig. 3.4. Cilindrii, piste, sectoare

Mai mult pentru dischete, structura logică este aceeași cu structura fizică. În cazul discului hard, la începutul acestuia (în primul sector fizic, fața 0, cilindrul 0) se află o *tabelă a partițiilor,* care conține informații privind organizarea volumului. Primul sector logic începe după primul sector fizic (ocupat de tabelă) și, în general, ordinea sectoarelor logice nu este aceeași cu a sectoarelor fizice.

În timp ce o dischetă nu poate fi utilizată decît cu un SO particular, un disc hard poate suporta mai multe SO. O tabelă a partițiilor aflată în primul sector fizic de disc (fig. 3.5.) definește, pe de o parte, unul sau mai multe drivere de discuri logice pe un același disc fizic iar — pe de altă parte — sistemul de operare pentru fiecare driver.

Tabela partițiilor începe la adresa 1BEH a primului sector fizic, are o lungime de 64 octeți și se termină cu secvența 55H AAH. Cei 64 octeți sînt împărțiți în blocuri de cîte 16 octeți, fiecare bloc folosind la păstrarea datelor de identificare a unei partiții. Dintre cele 4 partiții care pot fi definite într-un disc, una este *partiția activă,* din care se poate încărca sistemul de operare.

Din punctul de vedere al programatorului, datele sînt citite și scrise la nivel logic. Unitatea logică de acces la disc este *cluster-ul.* Un cluster corespunde numărului minim de sectoare logic adresabile pe o unitate de disc sau dischetă. Acest număr, ce depinde de capacitatea discului, variază între 1 și 2 la dischete și între 4 și 16 la discul dur (chiar 64 sectoare/cluster la discurile foarte mari).

Întrucît sistemul de operare citește și scrie datele numai în clustere întregi, rezultă că un fișier oricît de mic va ocupa un întreg cluster. Astfel, dacă vom crea un

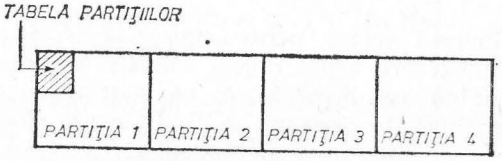


Fig. 3.5. Tabela partițiilor

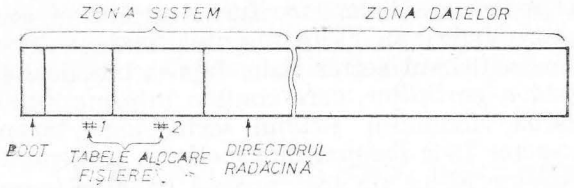


Fig. 3.6. Structura unui disc DOS

fişier de 20 octeţi pe un hard disc de 40 MO, el va ocupa în realitate 4 sectoare × 512 octeţi = 2 048 octeţi, deoarece pentru un astfel de hard disc, un cluster ocupă 4 sectoare. Comanda DIR va afişa pentru fişierul în cauză dimensiunea 20 octeţi, în timp ce prin comanda CHKDSK vom afla spaţiul real ocupat de fişier.

După formatare, atît dischetele cît şi discurile hard au aceeaşi organizare, excepţia constituind-o tabela partiţiilor cu care începe hard discul. Dar cum aceasta nu face parte din structura logică a discului, rezultă că orice disc logic din hard disc arată la fel cu o dischetă. Analizăm în continuare structura unui disc logic.

Sub MS-DOS discul logic are două zone : *zona sistem* şi *zona datelor*.

Aflată la începutul discului, zona sistem este rezervată sistemului de operare MS-DOS pentru păstrarea informaţiilor necesare gestionării zonei de date a discului. Mărimea zonei sistem depinde de tipul discului, dar oricum nu depăşeşte 1-2% din capacitatea discului. Restul spaţiului este ocupat de zona datelor, zonă folosită pentru memorarea datelor utilizatorului.

Zona sistem cuprinde : înregistrarea de autoîncărcare *(Boot record)*, tabela de alocare a fişierelor şi directorul rădăcină (fig. 3.6).

3.2.1. SECTORUL DE AUTOÎNCĂRCARE

Primul sector pentru un disc DOS este sectorul de autoîncărcare. Dacă discul este un disc sistem (a fost formatat cu opţiunea /S), el va conţine o copie completă a sistemului de operare DOS. În acest caz, în primul sector se găseşte un scurt program a cărui sarcină este să

înceapă procesul de startare a sistemului DOS din disc, citind din disc în memorie prima parte a programelor DOS. Primul sector conţine de asemenea blocul parametrilor BIOS (BPB).

Structura primului sector de disc este dată de tabelul 3.5. Întrucît această structură nu este definită de

Tabelul 3.5. STRUCTURA PRIMULUI SECTOR DE DISC

Adresa	Conţinut	Mărime (în octeţi)
00h	Instrucţiune de salt la programul de autoîncărcare	3
03h	Numele constructorului versiunea S.O. folosit la formatare (în ASCII)	8
0Bh	Numărul de biţi pe sector (BPS)	2
0Dh	Numărul de sectoare pe cluster (SPC)	1
0Eh	Numărul de sectoare rezervate (RS)	2
10h	Numărul de copii pentru FAT (CF)	1
11h	Numărul maxim de intrări în directorul rădăcină (D) 64 — dischete 5" 1/4, 160 Ko 112 — dischete 5" 1/4, 360 Ko şi 3" 1/2, 720 Ko 224 — dischete 5" 1/4, 1,2 Mo şi 3" 1/2, 1,44 Mo 512 — hard disc de 32 Mo	2
13h	Numărul sectoarelor din discul logic (total, TS)	2
15h	Tipul discului (TD) FEh — dischete, 5" 1/4, 160 Ko FDh — dischete, 5" 1/4, 360 Ko F9h — dischete, 5" 1/4, 360 Ko şi 3" 1/2, 720 Ko Foh — dischete, 3" 1/2, 1,44 Mo FBh — hard disc de 32 Mo	1
16h	Numărul de sectoare pentru FAT (SPF)	2
18h	Numărul de sectoare pe pistă (SPT) 8 — dischete, 5" 1/4, 160 Ko 9 — dischete, 5" 1/4, 360 Ko şi 3" 1/2, 720 Ko 15 — dischete, 3" 1/2, 1,44 Mo 17 — hart disc de 32 Mo	2
1Ah	Numărul capetelor de citire-scriere (NH)	2
1Ch	Numărul de sectoare rezervate sau ascunse (HS)	2

Microsoft, ea poate fi uneori puțin diferită. Oricum, în primul sector este înregistrată „amprenta" de identificare a discului și a tuturor zonelor din disc.

Pentru vizualizarea conținutului din primul sector al unei dischete am folosit programul DEBUG (fig. 3.7). Prima comandă (L 100 0 0 1) cere sistemului MS-DOS să

```
:\>debug
-l 0:100 0 0 1
-d 0:100
0000:0100  EB 3C 90 4D 53 44 4F 53-34 2E 30 00 02 02 01 00   .<.MSDOS4.0.....
0000:0110  02 70 00 D0 02 FD 02 00-09 00 02 00 00 00 00 00   .p..............
0000:0120  00 00 00 00 00 00 29 CA-18 66 2B 42 4F 4F 54 34   ......)..f+BOOT4
0000:0130  30 31 52 4F 58 20 46 41-54 31 32 20 20 20 FA 33   01ROX FAT12   .3
0000:0140  C0 8E D0 BC 00 7C 16 07-BB 78 00 36 C5 37 1E 56   .....|...x.6.7.V
0000:0150  16 53 BF 3E 7C B9 0B 00-FC F3 A4 06 1F C6 45 FE   .S.>|.........E.
0000:0160  0F 8B 0E 18 7C 88 4D F9-89 47 02 C7 07 3E 7C FB   ....|.M..G...>|.
0000:0170  CD 13 72 7C 33 C0 39 06-13 7C 74 08 8B 0E 13 7C   ..r|3.9..|t....|
```

Fig. 3.7. Conținutul sectorului de autoîncărcare (I)

încarce în memorie la adresa 100h, din unitatea 0 (unitatea A), începînd cu sectorul 0, 1 sector. Comanda *dump* afișează datele, în format hexazecimal.

Programul de autoîncărcare este lansat de încărcătorul ROM-BIOS care nu are nici o informație despre structura bloc-logică a discului. Prin urmare, el ajunge la primul octet din disc și îl execută ca pe o instrucțiune. Instrucțiunea de salt (primii 3 octeți), care permite programului să sară peste BPB, are una din următoarele două forme :

a) salt absolut (E9h) cu 2 octeți de deplasare : E9 XX XX

b) salt relativ (EBh), cu un octet de deplasare și o instrucțiune neoperativă — (NOP(9oh) : EB XX 90

Ținînd seama de structura prezentată în tabelul 3.5, vidajul din fig. 3.7 dă următoarele informații :

Instrucțiune de salt (3 octeți)	EB 3C 90
Nume și versiune (8)	MSDOS4.0
BPS(2)	512biți/sector
SPC(1)	2 sectoare/cluster
RS(2)	1 sector rezervat
CF(1)	2 FAT-uri
D(2)	112 intrări în rădăcină

TS(2)	820 sectoare/disc
TD(1)	dischetă, 5" 1/4, 360 K0 (FD)
SPF(2)	2 sectoare/FAT
SPT(2)	9 sectoare/pistă
NH(2)	2 capete de citire/scriere
HS(2)	0 sectoare ascunse

Octetul marcat cu un dreptunghi în fig.3.7 este octetul de început al programului de încărcare, program capabil să încarce restul sistemului de operare din disc.

În fig.3.8 este redat restul conținutului sectorului de autoîncărcare, inclusiv mesajele de eroare.

Este interesant de știut că sectorul de autoîncărcare există și la discurile care nu sînt de tip discuri sistem. În acest caz, sarcina acestui sector este de a semnala eroare la încercarea de încărcare a sistemului de pe un disc fără sistem de operare.

Există o clasă întreagă de viruși care infectează sectorul de autoîncărcare, cum ar fi Brain, Bouncing Ball (sau Ping Pong), virușii Stone ș.a.

```
0000:0180  89 0E 20 7C A0 10 7C F7-26 16 7C 03 06 1C 7C 13   .. |..|.&.|...|.
0000:0190  16 1E 7C 03 06 0E 7C 83-D2 00 A3 50 7C 89 16 52   ..|...|....P|..R
0000:01A0  7C A3 49 7C 89 16 4B 7C-B8 20 00 F7 26 11 7C 8B   |.I|..K|. ..&.|.
0000:01B0  1E 0B 7C 03 C3 48 F7 F3-01 06 49 7C 83 16 4B 7C   ..|..H....I|..K|
0000:01C0  00 BB 00 05 8B 16 52 7C-A1 50 7C E8 87 00 72 20   ......R|.P|...r
0000:01D0  B0 01 E8 A1 00 72 19 8B-FB B9 0B 00 BE DB 7D F3   .....r........}.
0000:01E0  A6 75 0D 8D 7F 20 BE E6-7D B9 0B 00 F3 A6 74 18   .u... ..}.....t.
0000:01F0  BE 93 7D E8 51 00 32 E4-CD 16 5E 1F 8F 04 8F 44   ..}.Q.2...^....D
0000:0200  02 CD 19 58 58 58 EB E8-BB 00 07 B9 03 00 A1 49   ...XXX.........I
0000:0210  7C 8B 16 4B 7C 50 52 51-E8 3A 00 72 E6 B0 01 E8   |..K|PRQ.:.r....
0000:0220  54 00 59 5A 58 72 C9 05-01 00 83 D2 00 03 1E 0B   T.YZXr.........
0000:0230  7C E2 E2 8A 2E 15 7C 8A-16 24 7C 8B 1E 49 7C A1   |.....|..$|..I|.
0000:0240  4B 7C EA 00 00 70 00 AC-0A C0 74 29 B4 0E BB 07   K|...p....t)....
0000:0250  00 CD 10 EB F2 3B 16 18-7C 73 19 F7 36 18 7C FE   .....;..|s..6.|.
0000:0260  C2 88 16 4F 7C 33 D2 F7-36 1A 7C 88 16 25 7C A3   ...O|3..6.|..%|.
0000:0270  4D 7C F8 C3 F9 C3 B4 02-8B 16 4D 7C B1 06 D2 E6   M|.....M|.....
0000:0280  0A 36 4F 7C 8B CA 86 E9-8A 16 24 7C 8A 36 25 7C   .6O|.....$|.6%|
0000:0290  CD 13 C3 0D 0A 4E 6F 6E-2D 53 79 73 74 65 6D 20   .....Non-System
0000:02A0  64 69 73 6B 20 6F 72 20-64 69 73 6B 20 65 72 72   disk or disk err
0000:02B0  6F 72 0D 0A 52 65 70 6C-61 63 65 20 61 6E 64 20   or..Replace and
0000:02C0  70 72 65 73 73 20 61 6E-79 20 6B 65 79 20 77 68   press any key wh
0000:02D0  65 6E 20 72 65 61 64 79-0D 0A 00 49 4F 20 20 20   en ready...IO
0000:02E0  20 20 20 53 59 53 4D 53-44 4F 53 20 20 20 53 59   .  SYSMSDOS  SY
0000:02F0  53 00 00 00 00 00 00 00-00 00 00 00 00 00 55 AA   S.............U.
```

Fig. 3.8. Conținutul sectorului de autoîncărcare (II)

Întrucît sectorul de autoîncărcare ocupă numai 512 octeți, un virus care infectează acest sector se va instala în întregime în sector numai dacă are o lungime mică (de exemplu, virușii Stone). Altfel va trebui să se împartă în două părți : „capul" virusului se instalează în sectorul boot, iar „corpul" virusului în alte sectoare, de obicei mai puțin folosite, pe care apoi le va declara ca sectoare defecte. Prin aceasta își asigură o ședere netulburată în sistem.

La autoîncărcarea sistemului de operare de pe un disc al cărui sector este infectat, derularea procesului de încărcare va fi modificată în sensul că primul va fi încărcat în memorie programul virus (prin simpla modificare a instrucțiunii de salt la programul de autoîncărcare, spre exemplu) și numai după aceea programele sistemului. O dată activat, virusul va infecta cu ocazia primei operații de citire/scriere orice disc care va fi introdus mai tîrziu în calculator.

Chiar și comenzile care fac numai citiri în disc (de exemplu, DIR sau CHKDSK) vor fi precedate de copierea virusului pe noul disc.

Un utilizator atent poate sesiza creșterea timpului la autoîncărcarea și, respectiv, în celelalte operații cu discul.

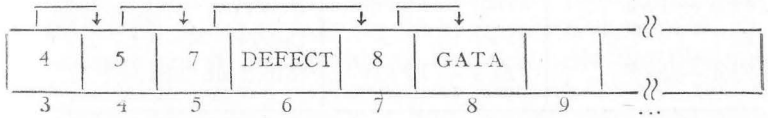
3.2.2. TABELA DE ALOCARE A FIȘIERELOR

Sectorul de autoîncărcare este urmat imediat de tabela de alocare a fișierelor (FAT — File Allocation Table). În această tabelă este înregistrată starea fiecărui cluster din disc : liber, alocat, rezervat, defect. Atunci cînd un fișier este înscris pe disc lui îi vor fi alocate clustere dintre cele libere. Primul cluster alocat fișierului se numește *cluster de început*. Toate clusterele care vor fi alocate fișierului vor forma o listă înlănțuită, listă înscrisă în FAT.

Cum funcționează această înlănțuire ? În FAT există o intrare pentru fiecare cluster. Numărul clusterului de început afectat fiecărui fișier creat sub DOS este memorat într-un cîmp din director. Pentru un fișier oare-

care, MS—DOS citește numărul clusterului de început din director. În FAT, intrarea corespunzătoare acestui cluster va conține numărul celui de-al doilea cluster din fișier ; intrarea corespunzătoare celui de-al doilea cluster va conține intrarea celui de-al treilea cluster din fișier ș.a.m.d. Cînd se ajunge la ultimul cluster din fișier intrarea din FAT corespunzătoare lui nu va conține numărul unui nou cluster, ci va conține un cod special care să simbolizeze sfîrșitul fișierului.

E x e m p l u : Pentru fișierul TEST.DOC, în director este precizată mărimea de 4 187 octeți și clusterul de început 3. Fișierul va ocupa 4 187 octeți/(512 octeți/sector x 2 sectoare/cluster) = 4,09 clustere. Întrucît clusterul este unitatea de alocare rezultă că fișierul va ocupa 5 clustere.



Clusterul 6 nu a fost alocat întrucît este deteriorat. El va conține un cod prin care MS—DOS este informat despre acest lucru.

Observație : În FAT înlănțuirea este unidirecțională : se poate găsi clusterul următor, dar niciodată clusterul anterior.

Numărul clusterelor pe care DOS le poate adresa este limitat de mărimea tabelei FAT și de mărimea fiecărei intrări din tabelă.

La versiunile DOS anterioare versiunii 3.0, mărimea fiecărui element din FAT era de 12 biți (3 intrări/2 baiți), asigurîndu-se astfel adresarea a 4 096 clustere (2.12). Începînd cu versiunea 3.0, un element din FAT poate avea, funcție de tipul discului, 12 sau 16 biți. Pentru dischete și hard discuri mai mici de 16 MO se utilizează FAT cu 12 biți, iar pentru hard discuri mai mari ; un element din FAT are 16 biți, fapt ce asigură adresarea a 65 536 clustere (la versiunile 3.X, pentru compatibili-

tate cu OS/2 v.1.o, tabela FAT nu are decît 16 384 intrări și 4 sectoare/cluster, ceea ce a limitat capacitatea de adresare la 32 Mo).

Deci FAT este o simplă tabelă de numere, cu cîte o intrare pentru fiecare cluster. Codurile trecute în fiecare dintre intrări și semnificația acestora sînt date în tabelul 3.6.

Tabelul 3.6. SEMNIFICAȚIA CODURILOR DIN TABELA DE ALOCARE A FIȘIERELOR

FAT cu 12 biți	FAT cu 16 biți	Semnificație
000	0000	Liber (nealocat)
002 — FEF	0002 — FFEF	Clusterul următor din lanț
FF0 — FF6	FFF0 — FFF6	Rezervat
FF7	FFF7	Defect
FF8 — FFF	FFF8 — FFFF	Sfîrșit de lanț

Pe cît este de simplă tabela de alocare a fișierelor, pe atît este de importantă. În esență, rolul tabelei este de a oferi sistemului DOS un spațiu distinct și separat în care să urmărească alocarea spațiului pe disc. Înregistrarea separată a listelor de înlănțuire asigură o protecție față de eventuale incidente (ștergerea din greșeală a unui fișier, de exemplu). Dacă FAT este distrusă, datele din disc nu mai pot fi recuperate și vor fi pierdute. Avînd în vedere importanța și „sensibilitatea" acestei zone, DOS păstrează mai multe copii de siguranță pentru FAT. În mod obișnuit, discurile au 2 copii ale FAT. DOS lucrează cu primul FAT, dar actualizează constant și pe a doua, astfel încît rămîn identice.

Toate aceste lucruri sînt foarte bine cunoscute de autorii virușilor. Pentru a-și masca prezența în disc, virușii poziționează de regulă intrările din FAT corespunzătoare sectoarelor ocupate de ei, ca sectoare defecte (FF7 sau FFF7). Pentru a-și realiza scopurile, alterarea octeților din FAT este o cale demnă de atenție pentru viruși.

3.2.3. DIRECTORUL RĂDĂCINĂ

Ultima parte a zonei sistem este ocupată de *directorul rădăcină*. Sub MS—DOS fișierele sînt grupate pe disc în directoare și subdirectoare. Directorul de bază este numit și directorul rădăcină ; el este prezent la toate discurile în timp ce subdirectoarele sînt opționale.

În director sînt înregistrate informații despre fișierele care-l compun. Fiecare fișier are în directorul rădăcină o intrare de 32 octeți. Spre deosebire de fișiere, directorul are o adresă fizică prestabilită pentru fiecare tip de disc, astfel că MS—DOS știe exact unde începe și unde se termină acesta.

Lungimea directorului rădăcină fiind fixă, există un număr maxim de intrări într-un director (tabelul 3.5, octetul 11 h). Acest număr nu dă și numărul maxim de fișiere care pot fi înregistrate în disc. Intrările din directorul rădăcină pentru subdirectoare (nu este impusă nici o restricție asupra lungimii și deci asupra intrărilor unui subdirector) fac ca numărul maxim de fișiere să fie limitat numai de capacitatea de memorare a discului și bineînțeles de dimensiunea fișierelor. Structura unei intrări în directorul rădăcină este prezentată în tabelul 3.7.

Cei doi octeți pentru timpul ultimei modificări (octeți 16h — 17h) au următoarea structură :

ORE					MINUTE						SECUNDE/2				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Întrucît pentru secunde sînt alocați 5 biți, vor putea fi reprezentate numai 32 valori distincte. Ca urmare, se păstrează numărul secundelor împărțit cu 2.

Structura octeților pentru dată este următoarea :

ANUL							LUNA				ZIUA				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Tabelul 3.7. STRUCTURA UNEI INTRĂRI ÎN DIRECTORUL RĂDĂCINĂ

Adresa	Conținutul	Mărimea (octeți)
ooh	Numele fișierului (8 caractere ASCII maxim)	8
o8h	Extensia fișierului	3
oBh	Biți de atribute o1h — acces numai în citire (read-only) o2h — ascuns (hidden) o4h — sistem o8h — identificator de volum 10 h — subdirector 2oh — arhivabil 4oh — (neutilizat) 8oh — (neutilizat)	1
oCh	(zona rezervată)	10
16h	Ora ultimei modificări a fișierului	2
18h	Data ultimei modificări a fișierului	2
1Ah	Primul cluster al fișierului	2
1Ch	Mărimea fișierului (în octeți)	4

Poate fi înregistrată orice dată între 1 ianuarie 1980 și 31 decembrie 2099. Ecuațiile folosite pentru DATE și TIME sînt următoarele :

DATE = ZIUA + 64 × LUNA + 512 × (ANUL —1980)

TIME = SECUNDE/2+32×MINUTE+2 048×ORE

Caracteristica tipurilor de viruși care infectează fișierele EXE și/sau .COM (exemple : TYPO, 640 K, Jerusalem, April 1th) este că modifică lungimea fișierului infestat, incluzînd la valoarea înainte de infectare și mărimea programului virus, dar data și timpul rămîn neschimbate. Cel puțin pentru COMMAND.COM este bine să-i cunoaștem dimensiunea și din cînd în cînd să verificăm respectarea ei pentru depistarea rapidă a unui virus.

Observație : Atunci cînd un fișier este șters, el nu este șters fizic din zona datelor de pe disc. MS—DOS

marchează ștergerea fișierului numai în zona sistem și mai exact în directorul rădăcină și în FAT. Astfel, în directorul rădăcină primul byte din intrarea corespunzătoare fișierului este setat la valoarea E5h (restul elementelor rămîn neschimbate). În FAT, toate intrările aferente fișierului vor fi marcate ca libere (000h). Luînd în considerare cele prezentate, pot fi concepute programe utilitare pentru recuperarea fișierelor șterse din greșeală (de pildă, Quick Unerase al lui Norton). I se cere utilizatorului să precizeze prima literă a fișierului (care a fost ștearsă) apoi sînt căutate în FAT intrările puse cel mai recent pe zero. Pe baza informațiilor din director, (mărimea fișierului), utilitarul va afla cîte clustere au fost alocate fișierului.

3.3. FORMATAREA DISCURILOR ȘI DISCHETELOR

Prin formatare se pregătește suportul pentru a fi în măsură să stocheze datele. Formatarea presupune două niveluri : formatarea fizică sau la nivel inferior ; formatarea logică sau la nivel superior.

Prin *formatare fizică* discul este pregătit din punctul de vedere hardware, el putînd fi ulterior folosit de orice sistem de operare. Pe parcursul acestei operații discul este împărțit în piste și fiecare pistă este divizată într-un număr egal de sectoare. Pentru fiecare sector este înscris, la intervale uniforme pe pistă, un antet de identificare *(ID sector header)*. Poziția de început a unui astfel de antet nu poate fi modificată de nici un sistem de operare. Urmează un număr unic pentru fiecare sector, număr care va fi utilizat de controlerul de disc pentru asigurarea poziționării corecte a capetelor de citire-scriere pe sectorul cerut. În timpul formatării fizice sînt identificate și sectoarele care prezintă anumite defecte.

Un cod magnetic specific, înregistrat în antetul de identificare al sectoarelor defecte, face ca aceste sectoare să nu fie folosite la formatarea logică.

Prin *formatare logică* discul este pregătit pentru a fi utilizabil sub un anumit sistem de operare.

De aceea, operația este realizată de sistemul de operare respectiv. Astfel, în cadrul formatării logice, pentru MS—DOS se asigură : regruparea sectoarelor în clustere ; inițializarea unor zone din disc (sectorul de autoîncărcare, tabelele de alocare a fișierelor, repertoriului directorului rădăcină) ; copierea fișierelor-sistem (dacă este cazul) ; înscrierea etichetei de volum. Formatarea logică a definitivează structura bloc-logică a discului.

Comanda FORMAT a sistemului de operare MS—DOS acționează în mod diferit pentru dischete față de discuri hard. Astfel, la formatarea unei dischete se realizează ambele formatări — fizică și logică. Întregul conținut al dischetei este șters și în fiecare octet al fiecărui sector se înregistrează secvența F6H. Datele existente pe dischetă înaintea formatării nu vor putea fi niciodată recuperate după formatare.

În cazul discului hard, comanda FORMAT realizează numai formatare logică. În timpul formatării datele existente pe disc nu sînt suprascrise, ceea ce face posibilă recuperarea acestora. Programul pentru formatarea fizică a discului hard este furnizat în memoria plăcii controlerului (formatarea fizică se face pentru un anumit controler, schimbarea controlerului implică reformatarea fizică a discului), în SETUP-ul din ROM sau pe dischete sub forma unor utilitare disc (de exemplu, Disk Manager, Speed Stor, Spin Rite).

Ținînd seama de cele prezentate, rezultă că formatarea unui disc hard presupune parcurgerea mai multor etape :

1. Formatarea fizică ;

2. Partiționarea discului în mai multe unități logice, scop în care se folosește comanda FDISK MS—DOS. *Atenție* : FDISK poate crea, șterge sau adăuga numai partiții DOS. Nu poate șterge o partiție non-DOS ;

3. Formatarea logică a unei unități prin comanda FORMAT.

4. Încărcarea opțională în disc a fișierelor IO.SYS (BIOS MS—DOS), MSDOS.SYS (nucleul MS—DOS) și COMMAND.COM (interfața utilizator).

4.

STRUCTURA ȘI FUNCȚIONALITATEA UNUI PROGRAM DE VIRUSARE

Existența unei mari varietăți de viruși informatici impune fără doar și poate nu numai o sistematizare a lor, ci și identificarea părților lor comune. Acest lucru se impune chiar și atunci cînd modalitățile de pătrundere a virușilor într-un sistem informatic, precum și manifestările specifice sînt dintre cele mai diferite.

4.1. MODULUL DE IMPLANTARE

Să schițăm mecanismul de „implementare" folosit de un virus pentru a putea pune în evidență modulele funcționale din care trebuie să se compună.

În primul rînd, el trebuie să aibă capacitatea de a se „grefa" pe un alt program executabil. Acest modul funcțional joacă același rol pe care îl are în cazul virusului biologic proteina virală, care protejează informația virală conținută în acidul nucleic (ribonucleic-ARN sau dezoxiribonucleic-ADN) și facilitează tranzitarea acestuia între celule. De ce este necesară această funcție în cadrul virusului informatic ? Cum poate fi ea îndeplinită în condițiile cele mai discrete cu putință ? Iată numai cîteva din întrebările fundamentale, față de care autorii programelor-virus nu sînt deloc străini.

Obiectivul de bază al virusului, servind unui scop distructiv, evident că nu va fi îndeplinit prin lansarea premeditată în execuție de către utilizator. Dacă, în general, autorul produselor-program diferă de beneficiarul lor, în

cazul virușilor informatici autorul coincide într-un anumit sens cu beneficiarul, iar proprietarul sistemului de calcul pe care evoluează un astfel de program suferă doar efectele nedorite ale acțiunii acestui program. Și totuși ! Am putea oare să ne imaginăm și programe-virus, care cunoscute ca existență și mod de acțiune să fie folosite de către proprietarul unui sistem de calcul ca ultima șansă de a interzice accesul unui „outsider" (intrus) la datele dintr-un sistem prin autodistrugerea acestuia ? (Și iarăși ne situăm atît de aproape de limitele umanului !) Dacă da, am putea identifica astfel și o valență „pozitivă" pe care s-o atribuim acestei clase cu totul aparte și atît de mult hulită de produse informatice !

Așadar, în mod obișnuit, un virus nu beneficiază de o lansare proprie în execuție. În ultimul timp, în pofida unui trafic clandestin cu produse-program, lumea informaticienilor și neinformaticienilor a devenit destul de circumspectă la execuția unor produse-program despre care nu se știe nimic sau aproape nimic ; sau dacă le execută, atunci o fac pe sisteme izolate și luînd toate măsurile de precauție pe care o asemenea situație le impune.

Protecția împotriva folosirii neautorizate a produselor informatice a beneficiat așadar de un efect, cel puțin moral, de descurajare, ca urmare a apariției și dezvoltării tehnicilor de virusare. Un lucru era știut, cu prețul mai multor ani de cercetare în domeniul biologiei : virusurile nu se multiplică și nu se întrețin decît pe medii vii. Să fi fost simplul experiment care a stat la originea acestor descoperiri sau mintea iscoditoare a omului a anticipat ceea ce ar fi trebuit să fie demonstrat ? ! Dacă cea de-a doua variantă pare și ea plauzibilă, atunci și informatica are șansa de a oferi omului un mediu de experimentare mult mai ușor de formalizat și studiat. Și poate, realitatea și oglinda ei vor merge și acum ca întotdeauna împreună, asigurîndu-și reciproc pretexte ulterior generatoare de progres !

Pentru a-și putea duce la îndeplinire misiunea distructivă virusul trebuie așadar să beneficieze de lansarea în execuție a altor programe și camuflat să-și realizeze și obiectivele proprii. El va alege practic, ca mediu de evoluție, orice program susceptibil de a fi încărcat în memoria principală și executat : programe de sistem și de utilizator, module de control periferice, procesoare de comenzi etc. Acest lucru explică de ce sînt afectate cu prioritate fișierele de tip EXE și COM, care conținînd cod executabil oferă un mediu propice virusului pentru a-și depune propriul cod executabil. Acțiunea malefică a unui virus se poate extinde și asupra celorlalte tipuri neexecutabile de fișiere, dar în acest caz ea urmărește în exclusivitate scopuri distructive, neglijînd aspectele referitoare la propagarea și proliferarea virusului.

Mai există și un alt motiv pentru care un program virus se implantează într-un program executabil. Pentru a putea pătrunde în zonele protejate ale sistemului, virusul are nevoie în mecanismul său intim de drepturi de acces pe care nu le are ; implicit, programul în care s-a implantat girează și drepturile de acces ale virusului, fără ca utilizatorul să știe despre acest lucru. Toate porțile fiindu-i deschise, virusului nu-i mai rămîne decît să-și pună în practică misiunea incriminată.

Sarcina de a se ataşa unui program executabil nu este deloc ușoară, mai ales cînd acest lucru se dorește a fi cît mai discret posibil și să treacă neobservat o perioadă de timp cît mai mare. Probabilistic vorbind, cu cît perioada scursă pînă la sesizarea prezenței virusului este mai mare, cu atît efectele nedorite produse de acesta sînt mai ample.

Așadar, să privim în detaliu, cu multă atenție întregul mecanism de fixare a virusului în fișierele care conțin cod executabil.

Cel puțin cîteva aspecte sînt fundamentale în îndeplinirea acestei funcții și se impun a fi studiate pe măsură. Cum sînt alese fișierele ce vor fi infectate ? Cum se evită implantarea virusului într-un program deja infectat ? Care sînt cele mai eficiente modalități de implantare pentru a nu fi sesizată de utilizator infectarea unui program ? Cum sesizează un virus prezența unui organism „viu" ? Pot exista viruși, cel puțin în formă latentă,

implantați în elementele așa-numitei lumi „moarte“ ? Iată doar cîteva din întrebările al căror răspuns trebuie să-l dăm în această etapă a analizei întreprinse de noi.

Alegerea tipului fișierelor ce urmează a fi afectate se face simplu, după informația de tip (EXE, COM, SYS. sau BAT) asociată fiecărui fișier în catalogul fișierelor.

Restricția de tip impusă la rularea fișierelor ce conțin cod executabil nu face decît să faciliteze la maxim munca de alegere a „celulelor“ ce vor permite dezvoltarea virusului informatic.

Renunțarea la obligativitatea tipului și transformarea acestei restricții doar într-o convenție de tip „implicit“ ar putea face cel puțin mai laborioasă munca de identificare a fișierelor care conțin programe executabile. Varietatea nominalizării tipurilor ar fi astfel lăsată la latitudinea imaginației utilizatorilor și orice varietate a fost dintotdeauna garanția unui progres sigur, chiar dacă prețul plătit pentru o evoluție prin diversitate a fost uneori destul de piperat.

La nivelul restricțiilor existente ne mai rămîne totuși o singură șansă, de altfel deloc neglijabilă : să suplinim noi această variantă, păstrînd în catalog fișierele sub cu totul alt tip, mai depărtat sau mai apropiat de convenția comună. Și cum măsurarea acestui grad de apropiere poate fi redusă în domeniul inteligenței umane comune, implementarea unor elemente de inteligență artificială echivalentă ei ar fi mult prea costisitoare, comparativ cu efortul uman depus pentru obținerea aceluiași efect. Am putea astfel schimba tipul EXE cu EXC, schimbare foarte sugestivă pentru om, dar care lasă cu totul „insensibil“ un program de tip virus. Pentru acesta din urmă gradul de apropiere al celor două tipuri, dacă nu se face cel puțin o analiză structurală neputîndu-se vorbi de una semantică, ar putea fi apreciat ca fiind la fel de mare ca între oricare alte două tipuri. Și pentru că s-au creat deja niște deprinderi în lumea utilizatorilor de produse informatice de ce nu am uza de ele, pentru că aceste deprinderi, ținînd tot de lumea inteligenței, nu pot fi suplinite nici măcar în parte de un program care se dorește cît mai mic cu putință. Utilizatorii înverșunați ai mini-

sistemelor ar propune probabil fără prea multă șovăială o schimbare a tipului EXE cu tipul TSK, a tipului BAT cu CMD, schimbări de nume de genul AUTOEXEC cu LOGIN sau SYSLOGIN și lista analogiilor funcționale mai mult sau mai puțin sesizabile pentru cele două tipuri de sisteme ar putea continua.

Schimbările la care se procedează, oricare ar fi ele, trebuie să constituie obiectul unui program specializat pentru a degreva utilizatorul de efectuarea unor renumiri de rutină, cu atît mai mult cu cît ele trebuie făcute cu o frecvență destul de mare. Un astfel de program trebuie să efectueze redenumiri în ambele sensuri : aducerea unui fișier la tipul EXE sau COM înainte de rulare și, respectiv, revenirea la tipul de camuflaj sub care se păstrează fișierele.

Maniera de implantare a unei secvențe virale într-un fișier de cod executabil este foarte diferită de la un virus la altul. Această secvență trebuie interpusă în fișier, astfel încît codul ei să fie executat întotdeauna, indiferent de ramura logică urmată în execuție de către programul gazdă. În acest scop, în general, codul viral este depus în partea inițială a unui program. Procesul de intercalare nu este deloc simplu. În primul rînd, un virus performant necesită o secvență de instrucțiuni destul de lungă. Grefarea ei pe un fișier executabil existent ar fi ușor sesizabilă de către utilizator, chiar fără eforturi orientate în acest sens. Propagarea virusului în toate fișierele conținînd programe executabile ar diminua substanțial spațiul liber pe disc, dacă nu chiar l-ar epuiza înainte de infectarea tuturor fișierelor susceptibile de a fi afectate. O astfel de situație face sigură depistarea virusului de către utilizator.

Așadar, se impune o implantare care să nu modifice dimensiunea programului-gazdă. Acest lucru este realizabil în cel puțin două variante. Una dintre acestea presupune înlăturarea unei părți din programul gazdă și înlocuirea ei cu o secvență virală. Această variantă are dezavantajul că modifică, în anumite situații, comportamentul programului-gazdă semnalînd indirect utilizatorului prezența unor disfuncționalități, în genul durerilor

de cap sau febrei în cazul organismelor infectate biologic.

În cazul unor programe (celor care conțin în prima parte modulul de monitorizare a tuturor funcțiilor, organul vital) această manieră de implantare scoate din folosință întregul program.

Cea de-a doua variantă este mult mai elevată. Ea presupune conservarea tuturor funcțiilor programului-gazdă, întîrziind la maxim momentul sesizării prezenței virusului de către utilizator. Cum este posibil acest lucru ?

Tehnicile de compactare joacă în acest caz un rol esențial. Între ele, algoritmul de compactare binară, datorat lui David Huffman și bazat pe utilizarea arborilor binari, prin performanțele sale ocupă fără doar și poate unul dintre cele mai importante locuri. Într-o primă fază programul gazdă este compactat după un algoritm de compactare adecvat fișierelor executabile, reducîndu-se spațiul ocupat de acesta, uneori chiar sub jumătate.

Spațiul economisit este folosit apoi doar parțial, pentru stocarea codului viral. Pentru a nu opera modificări în catalogul fișierelor, în această fază memorarea codului aferent virusului este completată cu umplerea spațiului rămas disponibil. De multe ori acesta va conține diverse mesaje care să intrige și mai mult pe utilizator în momentul efectuării unui *dump* asupra fișierului infectat. Pentru a asigura operaționalitatea programelor infectate, cel puțin pentru o perioadă de timp, această variantă de implantare comportă și o componentă pentru decompactare. Fiecare rulare a unui program infectat va începe cu o decompactare a fișierului infectat asigurîndu-se recunoașterea codului-obiect de către mașină.

Acest mecanism de rulare, valabil pentru o perioadă de timp stabilită de autorul virusului, sugerează și o cale de depistare a virusului, în momentul în care acesta se află în memorie, pe timpul rulării. Diferența dintre spațiul de memorie externă pe care-l ocupă ca fișier un program executabil și spațiul de memorie principală ocupat pe timpul rulării este un bun indiciu asupra prezenței procesului de compactare caracteristic virușilor inteligenți.

Și, cum totdeauna unde există două variante, există și o a treia, mixtă, care caută să combine în proporții acceptabile avantajele și dezavantajele variantelor de bază, și aici pot fi închipuite modalități de implantare dintre cele mai ieftine și eficiente totodată.

Poate fi exploatată, în acest sens, similitudinea funcțională, cel puțin parțială, a unor versiuni de taskuri care diferă sensibil ca dimensiune. Tocmai această diferență este folosită pentru implantarea unui virus, umplîndu-se fișierul pînă la dimensiunea fișierului echifuncțional, cu diverse ingrediente care nu joacă un rol important în mecanismul de funcționare a unui program virus.

Toate modulele funcționale ale unui virus trebuie să conțină instrucțiuni executabile. Ele au fost scrise însă într-o manieră ușor accesibilă programatorului în limbaje de programare și doar rareori date direct în cod mașină. Cine asigură procesul de compilare și linkeditare a secvenței virale ? Evident că o recompilare a taskului gazdă, astfel încît el să includă și secvențele parazite aparținînd virusului nu s-ar justifica, dacă nu ar fi chiar imposibilă. Să ne gîndim numai la faptul că doar rareori programele executabile sînt însoțite și de codul lor sursă sub forma unor fișiere conținînd instrucțiunile corespunzătoare în limbajele de programare evoluate, și cu atît mai mult în același catalog de fișiere și cu același nume de fișier cu cel executabil.

Prezența unui fișier de tip OBJ, conținînd codul obiect aferent unui program executabil, ar orienta probabil intenția spre a relinkedita programul gazdă, dar nu există certitudinea completitudinii modulelor-obiect ce intrau în structura programului și apoi lipsesc informațiile privitoare la bibliotecile necesare editării legăturilor. În plus, timpul necesar reconstruirii taskului ar da ușor de gîndit unui utilizator, sesizîndu-l că ceva nu este în regulă.

Sînt puține argumente, dar totodată suficiente, care să ghideze producătorii de viruși informatici spre crearea unor viruși autorelocabili. Forma executabilă a acestor viruși este intercalată printre modulele programului gazdă și ea reproduce în întregime structura unui fișier executabil. În acest sens, ea va conține atît dicționarul

de simboluri externe, cît și informațiile de relocare. Este bine știut că un program operează cu variabile și subunități de program, recunoscute la nivelul programului ca simboluri. Adresarea lor se face pe baza adresei atribuite la editarea legăturilor ; această adresă este relocabilă, fiind relativă la adresa zero atribuită începutului de task. În momentul rulării, adresa de început a zonei de memorie în care este încărcat taskul devine echivalentul adresei de început a taskului, toate simbolurile suportînd translatarea corespunzătoare față de acest punct. În cazul programelor virusate, față de modificările de adresă caracteristice unui task oarecare se mai impune o modificare datorată lungimii secvenței virale.

Atunci cînd implantarea virusului se face undeva în interiorul taskului gazdă și nu la începutul acestuia, trebuie avut în vedere ca funcția de relocare proprie virusului să opereze diferențiat pentru partea de program premergătoare virusului și, respectiv, pentru partea ce urmează secvenței virale.

Asigurînd și acest detaliu, un program infectat va putea fi încărcat și lansat ulterior în execuție chiar de sistem, ca un program obișnuit.

4.2. MODULUL DE PROPAGARE ȘI PROLIFERARE A VIRUSULUI INFORMATIC

Simpla cunoaștere a unor modalități de implantare a unor programe virus în fișiere executabile nu este suficientă pentru înțelegerea mecanismului de funcționare a unui virus informatic. Tot așa cum virușii biologici, spre deosebire de bacterii, nu se multiplică singuri, ci cu concursul celulei gazdă, virusul informatic va utiliza programul gazdă nu numai ca purtător, ci și ca mijloc de multiplicare, singur neputînd realiza acest lucru.

În afara scopului distructiv urmărit de virus, structura unui cod viral trebuie să asigure o transmitere a acestei secvențe către alte programe neinfectate încă și o proliferare a virusului în întregul sistem de calcul.

După ce a pătruns în sistem prin metoda „calului troian", un virus va urmări deci și propagarea sa în noul mediu. Este suficientă o singură rulare a programului vector pentru ca virusul să se implanteze în alte fișiere executabile. În general, încrederea utilizatorilor a fost încurajată prin furnizarea codului sursă al unor produse informatice. Nu rare au fost însă cazurile cînd și acestea s-au dovedit purtătoare de paraziți virali. Numărul mare al instrucțiunilor sursă, compactitatea limbajului de programare, lizibilitatea scăzută a programelor, lipsa unei autodocumentări sînt tot atîția factori favorizanți care atrag atenția asupra posibilității intercalării unor instrucțiuni virus printre cele valide. Șansa de a supraviețui și a se extinde rapid în noul sistem depinde de numărul programelor existente în cataloagele de fișiere și de ritmul de lucru al sistemului. Lansarea în execuție a mai multor programe într-un interval de timp dat permite și o rată mare de pătrundere a secvenței parazite în zonele neinfectate încă. Aspectul probabilistic al problemei reapare și prin modul în care virusul se „orientează" spre programele cu frecvență mare de rulare. Nu este deloc întîmplătoare deci atribuirea unor priorități în infectare, începînd cu programele de sistem a căror rulare este sigură, de cele mai multe ori și repetată în cadrul aceleeași sesiuni de lucru și pînă la un banal program de utilizator.

Nu ne mai miră deloc așadar infectarea cu premeditare a programelor de interfață cu claviatura, avînd în vedere că orice sesiune de lucru cu calculatorul începe și se termină printr-o apăsare de tastă ! Procesorul de comenzi COMMAND.COM, prin frecvența ridicată cu care este invocat, oferă un suport deosebit de activ în proliferarea virală.

N-a scăpat acestui tir nici programul de autoîncărcare a sistemului, care prin poziția sa cunoscută pe disc și prin prezența sa în topul programelor rulate a oferit dintotdeauna o tentație pentru creatorii de viruși. Sînt și programe a căror frecvență de rulare ține de specialitatea domeniului de activitate căruia i-a fost dedicat sistemul de calcul. Întrucît și sistemul de operare trebuia încu-

noștiințat asupra acestei specializări, nimic mai simplu decît să fie furată această informație și de către virus, descrierile din PATH devenind un ghid prețios în alegerea victimelor.

Așadar, fără scrupule, autorii programelor de virus s-au gîndit la exploatarea în luptă atît a caracteristicilor generale ale unui sistem, cît și a individualității fiecăruia. Această luptă pare să aibă ceva din lupta pentru existență, care este mult mai generală decît ne-o închipuiam noi la un moment dat. Să se fi gîndit oare marii producători de soft că numai lansînd pe piață viruși și antiviruși își vor putea proteja întrucîtva interesele, inițiind totodată o luptă continuă, generatoare poate și de progrese, în înțelegerea și evoluția informaticii ? La fel cum este greu de crezut că inventarea armelor dintre cele mai sofisticate s-a făcut în exclusivitate în folosul științei sau, dimpotrivă, a urmărit doar scopuri distructive.

Revenind la obiectul acestui capitol trebuie arătat că simpla copiere a virusului de la un program la altul lăsa prea mult supusă hazardului existența lui. Pentru a deveni mai eficient, mecanismul trebuia dublat de o sporire a șanselor de reușită. Nimic mai simplu decît să învățăm de la natură ; în cazul populațiilor în care se nasc mai mulți indivizi șansa de supraviețuire a speciei este considerabil mărită. Componenta de propagare a virusului va trebui să dobîndească astfel și funcții de proliferare. Cu prețul unui cod mașină mai extins, virusul își va asigura duplicarea în 2,4 sau mai multe copii ale propriei secvențe agresoare, la o singură rulare a programului purtător. Legile difuziunii în masă și legile creșterii populațiilor par să-și găsească și aici o strălucită aplicabilitate.

O variantă exponențială de proliferare își găsește astfel implementarea într-o componentă a virusului informatic, prin infectarea simultană a mai multor programe curate. Procesul trebuia totuși controlat. O proliferare prea rapidă ar fi pus sub semnul întrebării însăși existența virusului. Descoperirea lui prematură și cunoașterea unor reguli elementare de stopare a acțiunii lui ar fi însemnat eșuarea în îndeplinirea obiectivului distructiv. Atacul trebuie să fie, ca și în cazul semenului său biologic, lent și ascuns ; certitudinea reușitei lui nu trebuie contrapusă vitezei de realizare a obiectivului distructiv.

În lumea biologicului controlul proliferării face obiectul unor mecanisme foarte bine puse la punct. În informatică el trebuie îndeplinit de același modul de propagare și proliferare. Așadar, ce trebuie făcut ? În primul rînd, evitarea infectării unui fișier deja infectat. Implantarea virusului pe un program deja infectat nu s-ar justifica pentru că, pe de o parte, ar mări dimensiunea fișierului sporind probabilitatea sesizării virusului de către utilizator, — pe de altă parte — lansarea în execuție a programului este valorificată o singură dată de către virus, lucru realizabil de secvența virală deja existentă. În aceste condiții modulul de proliferare trebuie să aibă capacitatea de a discerne între un program infectat și unul neinfectat. Varietatea modalităților de implementare a acestei funcții este destul de mare. În general, un virus este capabil să recunoască infectarea celulei gazdă doar de către el sau de către o familie de viruși cu care este înrudit ; astfel, tehnica recunoașterii după o „amprentă virală" poate fi aplicată în rezolvarea acestui deziderat cu prețul a cîtorva baiți conținînd această componentă și plasați într-un loc ușor de regăsit, la începutul sau la sfîrșitul secvenței virale.

Alte mecanisme, cu nimic mai complexe decît acesta, uzează de faptul că utilizatorul, în maniera simplificatoare în care trebuie să vadă lucrurile, nu beneficiază în mod curent din partea sistemului de toate informațiile, unele minore, existente despre fișiere. Astfel, neglijarea secundelor din data creării sau ultimei actualizări a unui fișier poate reprezenta un suport pentru un virus de a face din el un indiciu al prezenței infecției virale, pe care numai el să-l trateze.

Deoarece la afișarea datei creării și actualizării unui fișier sînt înscrise doar ultimele două cifre ale anului, un program de virusare poate utiliza cu succes oricare din primele două cifre pentru a-și marca fișierele

deja infectate (însumarea a 100 ani, spre exemplu, la data ultimei actualizări).

Procedeul este cu atît mai eficient și rapid, cu cît nu presupune în faza de proliferare a virusului decît consultarea cataloagelor de fișiere și nu deschiderea fișierelor propriu-zise, în situația în care ele sînt deja contaminate. S-a crezut un timp că inhibarea scrierii pe un disc este suficientă pentru a opri propagarea unui virus în fișierele existente pe suport. Au apărut însă și viruși care afectează discurile inhibate soft la scriere, deturnînd informația de interdicție. Tot mai multe porți de intrare închise pînă nu demult au cedat în fața inteligenței agresorilor. Și de ce să n-o recunoaștem, vina aparține și creatorilor sistemelor de operare. Cu cît documentația privitoare la acestea a fost mai completă, în dorința unei utilizări cît mai complete și adecvate a sistemului, cu atît ea a constituit o sursă de nedisprețuit pentru autorii virușilor informatici, în găsirea unor modalități noi de pătrundere frauduloasă în sistem.

Să mai amintim de asemenea că, în materie de propagare a virușilor informatici, rețelele de calculatoare vor juca rolul avut de noile aglomerări umane în declanșarea unor epidemii de mari proporții. Modulul de propagare și proliferare se prezintă în acest context și ca o componentă instruită în vederea utilizării protocoalelor de transmisie în scopul satisfacerii tendințelor expansioniste.

Nu cîștigarea de spațiu urmărește această funcție, ci — pe de o parte — asigurarea unei șanse cît mai aproape de certitudine de a supraviețui, iar — de de altă parte — procurarea unor avantaje pe măsura acestei munci inteligente.

S-au întreprins chiar serioase studii privind viteza de propagare și proliferare a unui virus în sistemele mono și multiutilizator, cu sau fără facilitate de multitasking, cuplate sau nu în rețea. Orice activitate și deci și cea de proiectare a unor viruși eficienți se cere minuțios pregătită.

În sistemele cu multitasking, posibilitatea lansării simultane în lucru a mai multor programe a fost exploatată în sensul lansării reciproce în execuție a taskurilor virusate, contribuind la o proliferare care nu mai depindea acum decît de un singur impuls inițial. Și, ca și în cazul reacției de fisiune nucleară, proliferarea în lanț funcționa de la sine, creînd în sistem o adevărată febră a lucrului și abuzînd de toate resursele sistemului, pînă la epuizarea lor. Ca și semenii lor biologici, virușii informatici dispun de o independență suficient de mare ; „învățați" ce trebuie să facă o dată ce au pătruns într-un sistem neinfectat, li se dă drumul, scăpînd de cele mai multe ori de sub control, atît sub aspectul proliferării lor, cît și al pagubelor produse în sistem, comparabile cu ale unei bombe.

Un alt model experimentat pentru propagarea și proliferarea în sistem a virușilor informatici vizează „cuibărirea" acestora în blocul de autoîncărcare, prin aceasta asigurîndu-și încărcarea automată în memorie la oricare încărcare a sistemului.

4.3. MODULUL DISTRUCTIV

Ne-am lămurit pînă în prezent că virusul informatic implementează acea formă extremă a parazitismului practicată de virusul biologic, în care o secvență străină transmite ordine false, pe care programul gazdă nu le discerne în raport cu ale sale și le execută contribuind totodată și la proliferarea virusului. Din ce se compun acele ordine ? Ce anume mai realizează secvența virală în afara sarcinilor care-i privesc direct existența ? Un modul aparte pare a fi specializat în realizarea unor scopuri distrugătoare și în același timp profitabile pentru autorul virusului și nu pentru virusul însuși. Dacă privim sub aspectul obiectivului central al virusului am putea afirma fără să greșim că acesta ar putea fi considerat modulul principal al virusului. Pentru frumusețea actului programării și pentru analogia cu virusul biologic am tinde să credem că nu ; celelalte module, vizînd direct modul de existență a virusului, își dispută între ele titlul de importanță.

O întrebare și-o pune oricare utilizator care aude despre existența virușilor informatici : cît de mult pot distruge aceștia ? „Totul“ — ar răspunde cu o mîndrie nedisimulată creatorii de soft agresor ! „Deocamdată ne-am oprit la mai puțin, dar drumul ne este acum deschis, controlul în sistem ne aparține în exclusivitate“. Și trebuie să le dăm oarecare dreptate !

Dintr-o „generozitate“ uneori explicabilă, autorii fac din programele lor parazitare „artă pentru artă“, un soi de exercițiu al minții umane, testat în lumea reală. În aceste cazuri, aspectele distructive se reduc la mesaje ironice cu privire la prezența virusului. „Ha, ha, ha ! Tu ai un virus !“,„Este 1 Aprilie. Tu ai un virus“, „Lumea începe cu mine !“, „Eddie se află oriunde și oricînd !“ etc.).

Sînt bine cunoscute cazurile de viruși care se anunță singuri prin cîte o melodie sau printr-un desen animat, semănînd, mai mult sau mai puțin, panică printre utilizatorii de soft.

De cele mai multe ori deruta provocată în sistem generează blocaje, obligînd utilizatorul la reîncărcarea sistemului.

Alteori însă, aceste acțiuni sînt dublate și de grave alterări aduse sistemului, mergînd pînă acolo încît să-l facă inutilizabil.

Și cum probabil să distrugi este deseori mai ușor decît să construiești, efortul distructiv este minim ; este de ajuns să alterezi tabela de alocare a fișierelor pe disc și sistemul își va suprascrie singur fișiere, unul și același spațiu fiind alocat mai multor fișiere.

Întreaga structură de fișiere pe un suport tehnic de tipul discului dur sau dischetei devine de neutilizat printr-o simplă formatare a pistei de control. Orice recuperare a fișierelor devine anevoioasă și trebuie făcută prin programe specializate, care să reconstituie structura de regăsire standard.

O altă familie de viruși alterează fișierele din care provin programele în execuție curentă, suprascriindu-le cu altele, fără însă a le redenumi. Cînd utilizatorul solicită execuția acestui task, se execută de fapt un altul,

purtînd acest nume. Și dacă ne gîndim ce înseamnă execuția unui FORMAT în locul lui DISKCOPY, ne putem face o imagine asupra efectelor dezastruoase pe care un utilizator și le poate produce în această confuzie generală provocată de virus.

Alterarea sau distrugerea totală a datelor reprezintă unul din obiectivele foarte frecvente ale produselor virus. Nu de puține ori aceste soiuri de „răzbunări personale“ sînt înlocuite cu manipulări de date în interes personal. Am putea aminti aici implicațiile, deloc neglijabile, pe care le au modificările drepturilor salariale ale personalului, alterările premeditate aduse conturilor personale bancare, operarea unor transferuri de patrimoniu etc.

Acesta este motivul pentru care în țările dezvoltate cu o informatică bine reprezentată în toate sectoarele de activitate, o parte însemnată din aceste acțiuni constituie obiectul unor reglementări juridice stricte. Unele produse furnizate sub licență folosesc tehnici distructive, preluate ulterior și de virusuri, pentru a împiedica folosirea lor fără drept ; astfel, autodistrugerea unui program prin ștergerea fișierului conținînd codul său executabil reprezintă una dintre cele mai frecvente metode folosite în scopul amintit. O tehnică similară este utilizată și pentru a împiedica multiplicarea neautorizată a unui program ; aici copierea unui program este automat completată cu ștergerea originalului, determinînd ca la un moment dat prezența unui produs în sistem să fie unică.

Obiectivul modulului distructiv poate fi și unul destul de ascuns, legat de spionarea adversarului în procesul concurențial. Drepturile de acces cîștigate de virus pe seama programului gazdă pot reprezenta ele însele un tip aparte de efect distructiv urmărit prin virușii informatici și valorificat la maxim de către autorii lor.

Generalizarea rețelelor de calculatoare a facilitat o circulație deosebit de facilă și rapidă a informațiilor, dar nu fără riscuri în domeniul protecției și securității lor. Circulația prin rețele s-a dovedit la fel de eficientă și pentru produsele-program. Ea a permis însă și substragerea unor produse. Furtul de programe în vederea ne-

gocierii lor sau contrafacerii lor a devenit unul dintre obiectivele urmărite cu predilecție prin crearea de viruși informatici. Paginile revistelor de specialitate abundă în articole referitoare la cei care trăiesc dintr-un comerț înfloritor cu soft interceptat fără permisiune. Profesionalismul celor care se ocupă de astfel de lucruri, de cele mai multe ori tineri, reprezintă un obstacol dintre cele mai abile ce se interpune în calea depistării autorilor. Ca o ironie a soartei, tot sistemele de calcul sînt cele care păstrează gestiunea distribuitorilor clandestini de soft, cifrat bineînțeles, facilitînd valorificarea lui.

Simpla interdicție de a nu dezvolta și mai ales de a nu contribui la difuzarea virușilor din această categorie nu a dat rezultatele scontate.

Dacă la început efectul distructiv era mai puternic orientat spre deteriorarea suporților externi de informație, precum și spre ocuparea avansată a resurselor de rulare, ulterior virușii au beneficiat de metode dintre cele mai iritante, de a afecta consola sau diferitele porturi seriale și paralele disponibile în sistem. Mixarea semnificației tastelor de la claviatură, inhibarea tastaturii, schimbarea destinației unor pachete de ieșire sînt numai cîteva din „mesajele" dezavuabile ce semnalează prezența virușilor informatici.

```
„amprenta"

subprogram infectare
   repetă pînă găsești fișiere neinfectate :
         citește la întîmplare fișiere din catalog
         dacă nu au deja „amprenta"  proliferează virus ;

subprogram distruge
   efect distructiv

subprogram semnal
   răspunde doar la stimul extern ;

program principal
   infectare
dacă semnal distruge ;
   •
   •
   •
```

Fig. 4.1. Schema generală de funcționare a unui virus simplu

4.4. MODULUL DE CAMUFLARE ȘI AUTODISTRUGERE

Dacă în formele primitive virușii își impuneau oarecum cu brutalitate mesajul purtător de dezastre, în formele lor evoluate abilitatea de a se impune a crescut, făcîndu-i mai de temut ca oricînd. O funcțiune nouă se contura tot mai mult, aceea de camuflare a acțiunii virusului, cel puțin pentru o perioadă de timp ; pînă la instaurarea unei stări favorabile în sistem, pînă la un moment propice pentru deconspirare, cînd procesul distructiv devenise ireversibil, totul trebuia să pară absolut normal. În foarte multe cazuri toată acțiunea virusului se desfășoară cu o discreție totală. Urmărirea unor obiective de genul procurării de avantaje materiale pe seama pătrunderii neautorizate într-un sistem de gestiune impune acordarea unor valențe cu totul deosebite tehnicilor de camuflaj, făcînd uneori imposibilă depistarea virusului.

Asigurarea perioadei de „incubație" se face printr-o serie de acțiuni orientate exclusiv spre ascunderea tuturor modificărilor intervenite în sistem datorită virusului. Într-o primă etapă s-a crezut că utilizatorul este singurul care putea sesiza astfel de modificări, iar din această cauză virusul urmărea cu precădere mascarea aparențelor exterioare. Un fișier afectat trebuie să-și conserve data creării și a actualizării pentru a nu stîrni suspiciuni. Dacă implantarea virusului s-a făcut cu modificarea dimensiunii fișierului afectat, funcția de camuflare trebuie să asigure interceptarea comenzilor de afișare a rezumatului despre fișiere, precum și prezentarea lungimii inițiale a fișierelor, nicidecum a celei reale. În momentul dezvoltării unor programe specializate în depistarea virușilor și controlul utilizării resurselor sistemului, virușii infor-

matici s-au adaptat și acestor restricții. Camuflarea și deghizarea lor se făceau nu numai prin prisma aspectelor exterioare, ci și a mecanismului intern de lucru.

Astfel, intervențiile virusului în scopul proliferării sau manifestării acțiunii distrugătoare erau între cele mai scurte, pentru a nu putea fi interceptate de utilitare specializate în afișarea monitorizării resurselor. Rezidența în memorie era abil ascunsă, nefiind de multe ori gestionată pe calea clasică, utilizată de celelalte programe.

Se opta mai bine pentru o încărcare la o adresă fixă de memorie (de cele mai multe ori în partea finală a memoriei, așa cum era ea văzută prin sistemul de operare), decît să se folosească programele specializate în administrarea memoriei. Cînd prin acțiunea virusului sînt încălcate condițiile elementare de securitate a unor informații protejate, virușii inteligenți interceptează întreruperile originale, generate de aceste evenimente, deturnîndu-le către rutine de tratare proprii virusului ! Cele mai abile sisteme de protecție a bazelor de date se văd astfel studiate în liniște și ușor, ușor cedează în față inteligenței invadatoare, „șperaclului" informatic.

Camuflarea are în acest caz sarcina asigurării timpului necesar adaptării virusului la condițiile specifice ale sistemului, cu ajutorul cunoștințelor cu care a fost dotat.

Deși zestrea virusologiei biologice începea să se contureze pe la 1900, următoarele 2-3 decenii au fost fundamentale pentru lupta de idei dusă în clarificarea originii și naturii materiale a virușilor. Sesizarea prezenței și explicarea efectelor provocate de o clasă aparte de viruși, virușii cu acțiune lentă, avea să fie dovedită mult mai tîrziu, fiind încununată cu un premiu Nobel pe anul 1963.

Modelul de virus lent, preluat și în informatică, a permis crearea unor viruși latenți și cu o perioadă mare de incubație, care își asigurau astfel o prezență îndelungată într-un sistem de calcul.

Este meritul lui Cohen de a fi semnalat pentru prima dată importanța deosebită a acestei tipologii de viruși informatici, capabili să recunoască semnale externe pentru declanșarea acțiunii lor în masă. Dintre cei mai uzitați stimuli externi în raport cu care se sincronizează virusul, amintim data zilei, prezența unui program catalizator etc.

Acțiunea lor lentă trebuie să asigure funcționarea relativ normală a sistemului pe o perioadă mare de timp, dar să mențină și o posibilă dependență a sistemului față de virus. În acest scop modulul de camuflare poate realiza criptarea fișierelor de date, garantîndu-se un timp folosirea lor absolut corectă, dar numai utilizînd programele de prelucrare virusate. Orice utilizare a datelor este precedată de o decriptare a lor prin programul de virus, fără ca utilizatorul să știe ceva despre acest lucru.

Cînd atacul se declanșează, algoritmul de criptare/decriptare este distrus, iar reutilizarea datelor prin tratare devine imposibilă. Se implementează astfel o formă de „mascare în timp", care o completează pe aceea în spațiu, ținînd de existența materială a virusului informatic.

Merită menționată în acest punct al demersului nostru asupra mecanismului de funcționare a virusului informatic și o clasă de viruși „domesticiți", utilizați ca „dispozitive antifurt". Aceștia însoțesc soft-ul furnizat sub licență și, prin teste periodice asupra structurii programelor, combinate cu facilități hard dedicate acestui scop, sesizează utilizarea neautorizată a softului protejat. Am putea aminti aici virusul din Viena, sau virusul Dunării, cum mai este el cunoscut, a cărui acțiune se declanșează doar cînd se lucrează cu programe furate, care, evident, nu sînt dotate cu elemente de blocare a mecanismului antifurt.

Din punctul de vedere al modului de camuflare a prezenței o clasă aparte de viruși informatici o reprezintă *virușii evolutivi*. Ei sînt dotați cu capacitatea de a-și modifica esențial comportamentul pe parcursul existenței lor în calculator. Crearea virusului evolutiv intervenea după avîntul luat în lupta de identificare a virușilor prin programele specializate de căutare. Suportul căutării era oferit de însăși amprenta virală, deja depistată în cazul unor viruși, de lungimea secvenței virale, precum și de efectele acțiunii lor.

Reacția de adaptare la această situație a fost crearea de viruși cu structură variabilă, fundamentată aleator și implicit și cu o manifestare variabilă, funcție de natura codului viral executabil, înglobat în structură.

Pe această cale au fost creați viruși apropiați ca funcționalitate și totuși diferiți comportamental, punînd în mare încurcătură creatorii programelor antivirus și de devirusare. Problema majoră care apărea în acest caz ținea, pe de o parte, de imposibilitatea identificării proprietăților unui virus, evidențiabile doar în cursul unei acțiuni doar pe baza existenței lui latente, iar — pe de altă parte — de sporirea dificultăților de identificare în condițiile unei variabilități de comportament în acțiune.

Și pentru ca problema să devină aproape de nesurmontat, au fost creați viruși care după o infectare masivă a sistemului pur și simplu se autodistrugeau.

Autodistrugerea sau retransformarea programului într-un soft non virulent urmărea îndepărtarea oricăror surse de informații susceptibile de a permite reversibilitatea procesului distructiv inițiat de virus. Atunci cînd ea se făcea doar pentru unele programe virusate, avea drept finalitate îngreunarea muncii de identificare a programului purtător prin care s-a contaminat inițial sistemul.

În general, părăsind programul gazdă, virusul contribuia nu numai la păstrarea secretului privind programul vector, ci implicit și la protejarea autorului programului agresor, ascunzînd totodată calea de pătrundere în sistem și asigurîndu-și astfel o retragere care să permită o eventuală revenire !

Există voci care susțin încă inutilitatea unor astfel de manevre subtile, atîta timp cît sistemele de calcul pot fi agresate cu ușurință. Trebuie reamintit însă că nu au fost dezvăluite încă toate obiectivele urmărite prin prezența unui virus în calculator, iar — pe de altă parte — legislația pare să-și aplece tot mai mult atenția asupra actelor de pătrundere nepermisă în sistemele de calcul.

5.

INFECTAREA CU VIRUȘI ȘI BOMBE LOGICE

5.1. CONTAMINAREA PRIN TRANSFER DE FIȘIERE

Deși data apariției primilor viruși informatici este greu de precizat, fiind situată probabil undeva în jurul anilor '70, perioada marilor „epidemii" pentru informatică începe după 1980. Abia acum se va pune serios problema abordării acestui curent care punea în umbră marile posibilități deschise de informatică în lumea întreagă.

Anul 1988 a însemnat o declarație fățișă de război, o unire a eforturilor individuale în stăvilirea avîntului fără precedent pe care-l luase softul de factură distructivă.

Presa, prin popularitatea pe care o făcea acestui fenomen, contribuia și ea la conturarea în linii mari a tipologiei virușilor informatici, la etalarea principalelor căi și mijloace de infectare. Atenția acordată căilor de pătrundere a virușilor în sistemele informatice este strîns legată de posibilitățile de limitare a propagării virușilor informatici de la un sistem la altul.

Una dintre primele căi de pătrundere și care probabil rămîne printre cele mai frecvente și în prezent are la bază utilizarea dischetelor, ca suporturi tehnice de informație. Comoditatea și frecvența transferului de programe și date de la un sistem la altul, prin intermediul dischetelor, a făcut din acest mijloc de cooperare între sistemele informatice o cale dintre cele mai preferate pentru înfiltrarea unor produse virulente în sistemele informatice.

Funcționează în acest domeniul principiul *tranzitivității* semnalat de Fred Cohen, conform căruia dacă există o cale de infectare de la sistemul *A* la sistemul *B*, precum și o altă cale de infectare de la *B* la *C*, atunci există implicit și o cale de contaminare a sistemului *C*, cînd *A* este contaminat.

Pătrunderea în sistem a unui program purtător de secvențe virale nu este însă și suficientă pentru manifestarea funcțiilor virusului. Este nevoie ca programul, copiat sau nu de pe dischetă, să fie și lansat în execuție ; altfel el va rămîne un cod „mort", fără a perturba în vreun fel buna funcționare a sistemului. Este însă suficientă o singură lansare în execuție a unui program virusat din dischetă, pentru ca virusul să se implanteze pe un alt program cu frecvența mare de execuție și totul să nu mai poată fi controlat.

Este evident acum motivul pentru care transportul unor fișiere de date prin intermediul dischetei nu prezintă pericole, atîta timp cît, o dată cu ele, nu se transferă și programe executabile.

Observația rămîne perfect valabilă și pentru transferul fișierelor prin alte căi (rețea, disc dur etc.).

Acțiunea multora dintre virușii moderni se bazează pe captarea întreruperilor standard ale sistemului și implementarea prin intermediul lor a unor funcții inexistente în sistem pînă la acel moment, dar apelabile de către programul viral, care are cunoștință despre existența lor.

Pentru blocarea procesului de infectare se impune așadar și un control riguros al tuturor întreruperilor ce apar în sistem, chiar cu prețul creșterii sensibile a timpului de execuție al programelor activate în sistem.

În ce privește celălalt sens de transfer, din sistem către dischetă, problemele care se pun sînt la fel de complexe. Dacă pentru primele generații de viruși se considera că o dischetă nu este expusă riscului contaminării atîta timp cît pe ea nu se scrie, ci doar se citește din ea, odată cu pătrunderea virușilor informatici nu numai în fișierele de tip EXE și COM, ci și în modulele de control periferic, lucrurile s-au schimbat fundamental ; chiar

și în timpul citirii de pe un periferic, aceste drivere se comportă similar oricărui program în execuție, astfel încît virusul dobîndește drepturi de acces pe dischetă, ca de altfel pe oricare alt periferic. Astăzi este unanim recunoscut că este suficientă execuția unei comenzi DIR A : sau pur și simplu A : pentru a provoca o contaminare cu virus a dischetei.

5.2. INFECTAREA SISTEMELOR IZOLATE

O problemă, destul de controversată un timp, se referea la imposibilitatea contaminării unui sistem informatic izolat. Fără doar și poate șansele de contaminare scad pe măsura creșterii gradului de izolare a sistemului ; imposibilitatea contaminării unui sistem complet izolat a fost însă definitiv spulberată. Astăzi se cunosc exemple în care introducerea programului de virus s-a făcut pur și simplu de la consolă. Dacă introducerea codului viral într-un limbaj de programare ar da de bănuit, fie prin înțelegerea de către ceilalți utilizatori a obiectivului programului, fie ca urmare a timpului destul de mare, pînă la aducerea acestuia la forma executabilă, prin compilări și editarea legăturilor, atunci se poate opta pentru introducerea lui rapidă chiar sub formă de cod executabil. Acest lucru presupune însă și introducerea unor coduri ASCII neimprimabile și deci fără un semn echivalent pe claviatură. Tasta ALT permite însă, prin combinație cu altele, generarea celei mai mari părți a codurilor necesare. Un program de sistem, fie chiar și COPY, poate veni în întîmpinare pentru a prelua acest cod de la tastatură și a-l transforma într-un fișier pe disc !

De altfel, această manieră de lucru este ceva mai generală, ea permițînd introducerea unui program de virus și de la un alt port de lucru, serial sau paralel.

Tehnica programului purtător rămîne cea mai răspîndită cale de pătrundere în sistemele informatice a acestui veritabil flagel, care sînt virușii informatici. Amploarea pe care au luat-o jocurile pe calculator stă la originea unei adevărate „gripe informatice" provocate de vi-

ruşi. Analiza atentă a căilor de pătrundere a acestora şi a mecanismului de răspîndire reprezintă, într-o primă etapă, un mijloc de identificare a metodelor de limitare a propagării de la un sistem la altul, pînă la elucidarea completă a funcţionalităţii virusului.

5.3. LIMBAJELE DE PROGRAMARE A VIRUŞILOR

O problemă care se pune foarte frecvent în legătură cu viruşii informatici priveşte limbajul de programare folosit pentru realizarea acestora. Un prim aspect care trebuie luat în considerare la alegerea limbajului de programare este necesitatea realizării unor viruşi de dimensiune redusă, avînd posibilitatea să se implanteze în fişiere executabile mici, prin suprascriere sau prin ataşare, fără a ieşi în evidenţă. Din acest punct de vedere, se preferă limbajele care asigură o mare compactitate a codului executabil. Limbaje ca asambloarele C sînt preferate în acest sens.

Limbajele de asamblare mai oferă un mare avantaj : apropierea de codul maşină permite o exploatare maximă a facilităţilor sistemului de calcul. La acest nivel, toate sistemele de protecţie de factură soft sînt susceptibile de a fi depăşite, chiar executivul sistemului fiind invocat de cele mai multe ori pentru a escalada aceste restricţii. În plus, la nivelul limbajului de asamblare resursele de memorie şi timp de rulare pot fi atent drămuite. Toţi vectorii de întrerupere sînt direct adresabili, o parte din resursele soft de bază ale sistemului de operare devenind astfel disponibile.

Limbajele de programare evoluate, deşi conduc la viruşi de o lungime mai mare a codului executabil pentru realizarea unor funcţii date, oferă un mijloc foarte comod de a perfecţiona structura unui program de virus.

Sînt exploatate avantajele unei programări structurate, implementabile în limbaje ca Pascal şi C, dar practic, oricare alt limbaj, dacă este bine cunoscut, permite scrierea de viruşi informatici. În literatura de specialitate sînt publicate secvenţe virale realizate inclusiv în

limbajul BASIC. Puterea limbajului de a interacţiona cu sistemul este şi în acest caz implicată. Prin intermediul comenzii SHELL se obţin unele informaţii chiar de la sistem despre componenţa cataloagelor de fişiere, în vederea identificării victimelor, iar printr-o comandă APPEND un virus se poate implanta într-un fişier neinfectat pînă în acel moment.

Chiar şi sub un BASIC de tip interpretativ poate fi realizat un program de virus.

Recordul în materie, referitor la lungimea minimă a unui virus, aparţine pînă în prezent nu unui virus scris într-un limbaj de programare, ci unui virus compus din comenzi ale sistemului de operare ! El reprezintă un exemplu elocvent asupra modului în care facilităţile deosebit de puternice ale sistemului de operare sînt abil deturnate în scopul realizării unor manevre distructive.

Un astfel de virus execută în linişte doar cîteva comenzi adresate unor task-uri din sistem. Astfel, mai întîi redirecţionează consola spre perifericul nul pentru a nu deconspira numele task-urilor executate şi pentru a înlătura posibilitatea de întrerupere a execuţiei de la o consolă.

Se extrag, cu ajutorul comenzii DIR cu ieşire pe disc într-un fişier temporar, numele programelor susceptibile de a fi infectate.

Aceste informaţii sînt cerute în forma lor restrînsă ; excesul de informaţii puse la dispoziţie de DIR este înlăturat cu ajutorul unui editor de texte, apelat şi el pentru a executa comenzi prestabilite, ce fac obiectul unui fişier de comenzi indirecte.

Odată disponibilă lista fişierelor de infectat, ea va conţine informaţiile de plecare în procesul de proliferare a virusului. Este rîndul altui program de sistem de a prelua această sarcină. DEBUG, prin facilitatea de a lucra pe cod-obiect, se pretează foarte bine la această muncă. Prin intermediul unui fişier de comenzi indirecte, acest program reuneşte şi transformă fişierul victimă, astfel încît el să includă şi secvenţa virală. În acest scop sarcinile pot fi împărţite între utilitare de depanare, editoare şi comenzi ale sistemului de operare.

6.

IDENTIFICAREA ŞI CARACTERIZAREA VIRUŞILOR

6.1. VIRUS LATENT, VIRUS ... PRODUCTIV

La început spectrul restrîns al tipologiei virale nu ridica probleme deosebite în sesizarea prezenței unui „agresor" în sistem și în ceea ce privește identificarea acestuia. Ulterior, gama tot mai diferită a programelor parazite, precum și comportamentul lor foarte variat au impus alcătuirea unor modele pentru diagnosticare ; tehnici specifice erau puse la punct, în concordanță cu perfecționările de ultimă oră survenite în strategia virușilor informației. Utilitatea acestor metode de identificare și caracterizare a virușilor informatici este cu atît mai mare, cu cît acestea ar putea permite recuperarea unor elemente din sistemul infectat, de cele mai multe ori, de o însemnătate deosebită. Din acest punct de vedere, o importanță aparte o are clasificarea virușilor după *reversibilitatea acțiunii* lor. Acolo unde acest lucru este posibil, chiar cu prețul unei munci migăloase și de lungă durată, utilizatorii optează pentru această cale. O informație cît de mică despre un virus contează foarte mult în găsirea căii de tratare a fișierelor corupte. Mulți dintre autorii virușilor comunică chiar adresa lor pentru a beneficia de pe urma dependenței pe care au indus-o în sistemul de calcul.

În general, se acceptă că un sistem este contaminat în momentul în care funcții importante, uneori fundamentale, ale unor programe sînt afectate în timp, fără ca utilizatorul să fi intervenit modificîndu-le.

Cunoscînd acum principalele mecanisme de acțiune ale virușilor informatici putem să ne dăm seama că de multe ori este mai ușor de sesizat un virus, după desfășurarea cîtorva cicluri de infectare virală. *Caracterul latent* sau, dimpotrivă, *productiv* al unui virus este de asemenea esențial în reperarea prezenței virale într-un sistem de calcul. În timp ce virușii latenți sînt foarte greu de identificat, cei productivi, pe măsură ce acționează, devin tot mai ușor de identificat.

Trebuie făcută distincția aici între virușii care *distrug efectiv* și cei care doar *sugerează anumite anomalii* în sistem, fără ca acestea să existe de fapt.

În caracterizarea virușilor informatici se apelează la încadrarea lor într-o anumită generație, în raport cu modul în care își desfășoară activitatea, dacă afectează toate *tipurile de fișiere* sau numai unele tipuri, dacă este *rezident sau nu*, dacă utilizează *întreruperile standard* ale sistemului etc.

6.2. STRATEGII DE CĂUTARE

De multe ori, chiar identificarea căii posibile de pătrundere a unui virus în sistem poate furniza indicii cu privire la virus. Păstrarea unor relații cu alți posesori de sisteme de calcul similare și informarea reciprocă cu privire la anomaliile în funcționarea sistemului au menirea de a permite discernerea între ceea ce se datorează sistemului în general și ceea ce este rezultatul potențial al unui virus. O serie de factori favorizanți, de tipul utilizării unor programe provenind din surse ilegale, posibilitatea intervenției în sistem a unor persoane ce nu prezintă încredere etc. ne obligă la efectuarea unor controale periodice privind depistarea unor viruși informatici. În general, programele de scanare folosite în acest scop implementează atît *strategii de căutare orientată*, cît și *strategii generale* de identificare a unor viruși. Strategiile orientate sînt rezultatul încorporării în programe a unor informații cu privire la virușii cunoscuți pînă la acel moment : lungime, amprente, mod de acțiune, mecanism de proliferare.

Strategiile generale vizează comparații periodice ale programelor cu originalul (copie martor), gestionarea lun-

gimii fișierelor, interceptarea și monitorizarea tuturor task-urilor rulate în sistem, examene inopinate ale ocupării resurselor-calculator și altele.

Un control cu totul deosebit trebuie aplicat materialelor noi, intrate în sistem.

O cale destul de uzitată pentru identificarea și caracterizarea unor viruși o reprezintă studierea atentă a efectelor produse de aceștia. Acest lucru presupune conservarea pe cît posibil a sistemului infectat. Reacția de a șterge tot și de a lua-o de la capăt, pe care o are un utilizator nervos la constatarea că întregul sistem este infectat, contravine regulilor de bază ale comportamentului antivirus.

Nu trebuie uitat că modul cel mai eficient de a studia un program de virus îl reprezintă tocmai „disecarea" sa. Locul microscopului electronic este luat în cazul virușilor informatici de tehnicile de dezasamblare. Secvența virală tradusă în termenii unui limbaj de asamblare reprezintă pentru un specialist o modalitate deosebit de facilă pentru a-și cunoaște inamicul și a-l combate. Eventualele tratamente eficiente nu pot avea la bază decît rezultatele unei analize atente a structurii virusului, făcîndu-se astfel distincție între erorile reale și cele „mimate" sau sugerate prin diverse mesaje de eroare.

Faptul că un virus, odată existent în sistem, recunoaște fișierele deja infectate printr-o amprentă virală proprie, nu exclude riscul unei infectări multiple cu alte categorii de viruși informatici. Dacă avem în vedere și interferențele care se pot crea prin acțiunea simultană a mai multor categorii de viruși ne putem face o imagine despre dificultatea identificării corecte a unui virus și ireversibilitatea distrugerilor provocate de aceștia.

6.3. CARACTERISTICI DE BAZĂ ALE VIRUȘILOR

Funcțiunile principale ale unui virus informatic sînt generale, dar maniera de a le pune în aplicare este specifică fiecărui program în parte. Esențiale pentru aplicarea acestor funcții par cîștigarea dreptului de a scrie în zone protejate și capacitatea virusului de a culege informații despre structura sistemului informatic în scopul identificării indicelui de proliferare a infecției.

Maniera specifică în care virusul informatic își asigură funcțiile sale de bază permite delimitarea principalelor categorii de viruși.

Viruții care suprascriu porțiunea de la începutul programului sînt realizați pornindu-se de la ideea că primul lucru pe care trebuie să-l îndeplinească un program infectat, la rulare, este de a infecta alt program. Indiferent care va fi structura logică a programului infectat modulele specifice virusului informatic vor fi executate.

Dacă notăm cu *A* porțiunea conținînd amprenta virală, cu *V* nucleul virusului care asigură proliferarea, iar cu *D* modul distructiv al virusului, structura unui program infectat va fi :

A	V	D	program

Efectul distructiv se manifestă numai după proliferarea virusului în întregul sistem ! Prin suprascrierea unei zone din program cresc șansele apariției unor anomalii în funcționare, contribuind la deconspirarea virusului.

Viruții care nu suprascriu zone din programul purtător se implantează într-un fișier conținînd program executabil, mărind dimensiunea fișierului cu lungimea secvenței parazite. Este posibilă o implantare a virusului și fără supradimensionarea fișierului gazdă, prin comprimarea fișierului după un algoritm performant.

În cazul virușilor care nu suprascriu programul purtător, șansa depistării virusului în faza de proliferare este mică deoarece comportamentul programului infectat este perfect normal.

Viruții care se autodistrug dispun și de un modul de autosuprascriere, după încheierea misiunii de proliferare și distrugere în sistem. Dacă notăm cu *M* modulul care execută această funcție, printr-o mutare a primei părți dintr-o aplicație și aducerea unei părți a virusului în acest loc, putem reprezenta schematic principalele faze ale funcționării virusului informatic.

a) programul vector este purtătorul unui virus în forma lui completă :

A	V	D	M	program vector

b) virusul decupează o parte din fișierul executabil, egală cu mărimea cumulată a zonelor *A, V, D.* Zona decupată este atașată în coada programului, iar locul ei este preluat de o parte din virus ; în partea finală a programului este atașat și modulul *M* al virusului :

A	V	D	aplicație (II)	aplicație (I)	M
← *l* →				← *l* →	

c) virusul proliferează în tot sistemul ;
d) virusul exercită acțiunea distructivă ;
e) virusul își autodistruge mesajul distructiv, de proliferare și de identificare, prin activarea modulului *M :*

aplicație (I)	aplicație (II)	aplicație (I)	M

Chiar dacă, așa cum se observă, aplicația dispune în acest moment de o dublură a părții întîi, funcționarea sa este perfect normală. S-a pierdut însă orice informație care ar putea sta la baza recunoașterii virusului și eventual ar putea ajuta la recuperarea unor fișiere de date, afectate prin efectul distrugător exercitat de virus în fazele anterioare.

Codificările caracteristicilor de bază ale virușilor depistați pe microcalculatoarele IBM-PC sau pe cele compatibile cu acestea sînt : *1)* infectează discul hard (tabela de partiții) ; *2)* infectează boot-sectorul discului hard ; *3)* infectează boot-ul la dischete ; *4)* infectează fișiere overlay ; *5)* infectează fișierele .EXE ; *6)* infectează fișierele .COM ; *7)* infectează COMMAND.COM ; *8)* virusul rămîne rezident ; *9)* viruși care folosesc autocriptarea ; *10)* dezinfectator ; *11)* incrementarea dimensiunii programului infectat (cu cît crește mărimea programului) ; *12)* efectele produse.

Tabelul 6.1. PRINCIPALELE CARACTERISTICI ALE UNUI VIRUS

Nr. crt.	Nume virus	Caracteristicile virusului											
		1	2	3	4	5	6	7	8	9	10	11	12
1	CHAOS										MDISK	M	B, O, D, F
2	VIRUS—90		×	×							SCAN/D	857	P
3	OROPAX						×		×		SCAN/D	2773	P, O
4	4096				×	×	×	×	×		SCAN/D	4096	D, O, P, L
5	DEVIL'S DANCE						×		×		SCAN/D	941	D, O, P, L
6	AMSTRAD						×		×		SCAN/D	847	P
7	PAY DAY				×	×	×		×		M-JRUSLM	1808	P
8	DATACRIME II-B					×	×	×		×	SCAN/D	1917	P, F
9	SYLVIA/HOLAND						×		×		SCAN/D	1332	P
10	DO NOTHING						×		×		SCAN/D	608	P
11	SUNDAY				×	×	×		×		SCAN/D	1636	O, P
12	LISBON						×		×		SCAN/D	648	P
13	TYPO/FUMBLOE						×		×		SCAN/D	867	O, P
14	DBAGE						×		×		SCAN/D	1864	D, O, P
15	GHOST BOOT VERSION		×	×					×		MDISK	M	B, O
16	GHOST COM VERSION						×				SCANID	2351	B, P
17	NEW JERUSALEM				×	×	×		×		M-JRUSLM	1808	O, P
18	ALABAMA					×	×		×		SCAN/D	1560	O, P, L
19	YANKEE DODLE					×	×		×		SCAN/D	2885	O, P
20	2930					×	×		×		SCAN/D	2930	P
21	ASHAR			×			×		×		MDISK	N/A	B
22	AIDS										SCAN/D	overwrites	PROGRAM
23	DISK KILLER		×	×					×		MDISK	M	B, O, P, D, E
24	1536/ZERO BUG					×	×		×		SCAN/D	1536	O, P
25	MIX1					×			×		SCAN/D	1618	O, P

(continuare tabelul 6.1)

Nr. crt.	Nume virus	Caracteristicile virusului											
		1	2	3	4	5	6	7	8	9	10	11	12
25	DARK AVENGER				×	×	×	×	×		M—DAN	1800	O, P, L
27	3551/SYSLOCK					×	×			×	SCAN/D	3551	P, D
28	VACSINA				×	×	×		×		SCAN/D/A	1206	O, P
29	OHIO			×					×		MDISK	M	B
30	TYPO (BOOT VIRUS)		×	×					×		MDISK	M	O, B
31	SWAP/ISRAELI BOOT			×					×		MDISK	M	B
32	1514/DATACRIME II					×	×			×	SCAN/D	1514	P, F
33	ICELANDIC II					×			×		SCAN/D	631	O, P
34	PENTAGON			×							MDISK	M	B
35	3066/TRACEBACK					×	×		×		M—3066	3066	P
36	1168/DATACRIME B						×			×	SCAN/D	1163	P, F
37	ICELANDIC					×			×		SCAN/D	642	O, P
38	SARATOGA					×			×		SCAN/D	632	O, B
39	405						×				SCAN/D	overwrites	PROGRAM
40	1704 FORMAT						×		×	×	M—1704	1704	O, P, F
41	FU MAN CHU				×	×	×		×		SCAND/D	2083	O, P
42	1280/DATACRIME						×			×	SCAND/D	1230	P, F
43	1701/CASCADE						×		×	×	M—1704	1701	O, P
44	1704/CASCADE-B						×		×	×	M—1704	1704	O, P
45	STONED/MARIJUANA	×		×					×		MDISK P	M	O, B, L
46	1704/CASCADE						×		×	×	M—1704	1704	O, P
47	PING-PONG-B		×	×					×		MDISK	N/A	O, B

(continuare tabelul 6.1)

Nr. crt.	Nume virus	Caracteristicile virusului											
		1	2	3	4	5	6	7	8	9	10	11	12
48	DEN-ZUK			×					×		MDISK	N/A	O, B
49	VIENNA-B						×				SCAN/D	648	P
50	PING-PONG			×					×		MDISK	M	O, B
51	LENIGH							×	×		SCAN/D	overwrites	P, F
52	VIENNA/648						×				M-VIENN	648	P
53	JERUSALEM-B				×	×	×		×		M-JERUSL	1808	O, P
54	YALE/ALAMEDA			×					×		MDISK	M	B
55	FRIDAY 13 th COM						×				SCAN/D	512	P
56	JERUSALEM				×	×	×		×		SCAN/D/A	1808	O, P
57	SURIVØ3				×	×	×		×		SCAN/D		O, P
58	SURIVØ2					×			×		SCAN/D	1488	O, P
5	SURIVØ1						×		×		SCAN/D	897	O, P
60	PAKISTANI BRIAN			×					×		MDISK	M	B

* În tabelul 6.1 sînt dați 60 de viruși cu principalele lor caracteristici (coloanele numerotate cu *1-12* marcate prin × arată prezența caracteristicilor enumerate anterior).

În coloana *11*, prezența simbolului *M* indică mascarea alterării lungimii fișierului prin interceptarea comenzilor de afișare a acestora. În coloana *12*, notațiile *B, O, D, F, L, P* arată următoarele efecte produse de virusul respectiv : *B* — corupe și suprascrie în sectorul de autoîncărcare ; *O* — afectează sistemul prin programele ce se găsesc în execuție ; *D* — atacă fișierele de date ; *F* — formatează sau șterge parțial/total discurile ; *L* — distruge sau alterează direct/indirect fișierele de link ; *P* — distruge/alterează programul sau suprascrie fișierele : PROGRAM — sinonim cu *P*.

7.

TRATAMENT, DEVIRUSARE, IMUNIZARE

Așa cum un organism viu trebuie supus periodic unui examen medical atent, tot la fel și un sistem informatic necesită un control periodic al „stării sale de sănătate".

O măsură de prim ajutor care se impune în cazul sesizării unei infecții cu viruși informatici o reprezintă stoparea procesului de proliferare. Deoarece virusul informatic, comparabil întrucîtva cu cel biologic, nu are o viață „independentă", oprirea oricărei activități în sistem inhibă și propagarea infecției virale. Orice analize asupra sistemului vor fi făcute numai cu ajutorul unui sistem extern, sănătos.

În general, în lupta împotriva virușilor informatici se recunosc mai multe categorii de acțiuni : măsuri de profilaxie, controale periodice pentru depistarea infecțiilor, localizarea și distrugerea programelor care constituie focare de infecții, devirusarea sistemului, construirea unor mecanisme de tip imunitar și, evident, măsuri de ordin general privind educarea personalului.

În continuare sînt prezentate cîteva din aspectele mai importante și modalitățile lor de realizare concretă care intervin în lupta antivirus.

7.1. CUM SĂ NE PĂZIM DE VIRUȘI ?

1. Evitați folosirea dischetelor neverificate. La prima folosire a oricărui disc verificați dacă este de la un prieten sau de la altcineva. De un real folos vă poate fi programul SCAN.

2. Folosiți protecția fizică a dischetelor cînd nu intenționați să scrieți în ele.

Orice mesaj de forma : „*Write protect error writing — drive A :*" care apare după ce ați luat această măsură trădează o activitate suspectă, probabil a unui virus.

3. Utilizați comanda CHKDSK pentru verificarea memoriei la fiecare încărcare a sistemului. Absența cîtorva kbaiți din valoarea cunoscută pentru memorie ar putea indica prezența unui virus în memorie. Urmăriți cu grijă sectoarele marcate ca defecte mai ales pe dischete, întrucît față de discurile hard, la dischete sectoarele defecte apar mai rar.

4. Folosiți puține intrări în directorul rădăcină. Încercați să limitați directorul rădăcină la 32 fișiere, deoarece intrarea fișierului 33 începe la sectorul 103, sector alterat de virusul Stoned. În acest scop folosiți subdirectorii.

5. Utilizați unitatea logică D : în loc de C : pentru fișierele importante. Viruții obișnuiți infectează unitatea C. Comanda FDISK vă permite partiționarea discului hard în mai multe unități logice. Încărcați în C : soft-ul temporar sau în testare, iar în D : programele și datele format.

6. Păstrați programele separat de date. Este recomandabilă folosirea unor directori, subdirectori sau chiar a unor partiții diferite pentru date și programe. Se reduce riscul copierii unor viruși o dată cu datele.

7. *Backup your files.*

E bine să vă faceți un obicei din a vă „trata" fișierele. Este cea mai bună protecție pentru munca dumneavoastră.

7.2. DETECTAREA VIRUȘILOR ÎN SECTORUL DE AUTOÎNCĂRCARE

Avînd în vedere că dischetele au destul de rar sectoare defecte, pentru depistarea virușilor care atacă sectorul de autoîncărcare se recomandă verificarea dischetelor cu CHKDSK A : . Prezența a 3 KB în sectoare defecte poate fi un semn că discul este infectat cu virusul

Brain. Dacă sectoarele defecte ocupă 1 KB probabil că vina o poartă Bouncing Ball.

Confirmarea existenței virusului se poate face cu un utilitar, de genul PC Tools sau Norton Utilities. În acest scop se vizualizează sectorul de autoîncărcare, precum și sectoarele marcate ca defecte. Dacă sectorul de autoîncărcare este modificat și programul de autoîncărcare original a fost mutat într-un sector marcat ca defect dovada este făcută. În cazul în care sistemul a fost încărcat din discheta infectată cu un virus de tipul Brain, Bouncing Ball sau Stoned, comanda CHKDSK va afișa spațiul total al memoriei micșorat cu cîțiva KB.

Pentru discurile hard se impune verificarea atît a sectorului de autoîncărcare (primul sector logic), cît și a sectorului alocat tabelei de partiții (primul sector fizic). Folosește pentru aceasta un utilitar, cum ar fi Norton Utilities. Un disc infectat cu virusul Stoned va conține în primul sector fizic mesajul „*Your PC is now Stoned*". „Un sector de autoîncărcare sănătos" are la început secvența de litere MSDOS... sau PCDOS... Dacă lipsesc aceste litere, este posibil ca discul hard să fie infectat cu Bouncing Ball. Dacă sînt modificări atît în tabela partițiilor, cît și în sectorul de autoîncărcare înseamnă că doi viruși sînt prezenți (probabil Stoned și Bouncing Ball).

Observație : Dacă nu dispuneți de Norton Utilities pentru verificarea sectorului de autoîncărcare puteți utiliza utilizatorul sistemului DOS, DEBUG ( ).

7.3. PROGRAME DE DEVIRUSARE

Ca orice organism, și calculatorul poate trăi, bine sau rău, cu microbul în interiorul său. Dar cît și cum ? Totuși, mai devreme sau mai tîrziu, în urma uzurii fizice și morale, organismul cedează. Pentru a preîntîmpina fatalitatea este bine să consultăm un doctor, în cazul nostru un doctor... informatic care să ne prescrie rețete

de curățire / deparazitare / dezinfectare. Acest lucru este realizabil prin intermediul programelor de devirusare.

Se cunosc o multitudine de astfel de... specialiști. Dar niciodată nu putem afirma cu certitudine că tot ce am făcut noi este ultima soluție. Poate alții au ceva mai bun, mai eficient. De aceea vă sugerăm să urmăriți cîteva programe de devirusare, cu mențiunea că lista rămîne deschisă.

V-ANALYST

Utilizînd un algoritm al profesorului Michael Rabin (cîștigător al premiului Turing), un specialist israelian în securitatea datelor a realizat la societatea INFODIDACT programul V-ANALYST. Acesta permite depistarea tuturor modificărilor, voluntar sau involuntar, cu bună sau rea intenție, operate asupra tuturor tipurilor de fișiere și aplicațiilor (.EXE, .COM, .BAT).

Programul are trei module : *V-Immune* pentru prevenire, *V-Analyst* pentru detectare și *V-Unvirus* pentru dezinfectare.

Modulul *V-Immune* este rezident în memorie și ocupă 1 Koctet. Lista virușilor detectabili este revizuită trimestrial.

Modulul *V-Analyst* creează o listă codificată a tuturor fișierelor prezente pe o unitate de disc specificată, inclusiv sectorului de auto-încărcare și a fișierelor sistem. După ce aceste informații au fost înregistrate, programul verifică la începutul fiecărei sesiuni de lucru, integritatea fișierelor înregistrate în listă. În cazul depistării unor discordanțe, se emite un semnal de alarmă. Utilizatorul poate autoriza actualizarea listei cu noile informații sau lansarea modulului *V-Unvirus* pentru reconstruirea fișierelor executabile infectate.

Antivirus VIENNA, antivirus DANUBE

Există deja trei programe care sînt capabile să recunoască amprentele 909090h sau E9XXXXF în debutul fi-

șierelor sau amprenta 30/31 mm în repertoriu. Grație acestor programe vom putea detecta o serie de viruși cum sînt cei ai Vienei și Dunării.

Antivirusul ISRAEL

Virusul *Israel* bîntuie pe PC-urile din Europa. După toate informațiile noastre există societăți care explorează modul de acțiune al acestui virus și al descendenților săi.

Antivirus pe Amigo

În lumea lui Amiga există o mare varietate de viruși. Cei mai cunoscuți sînt SCA și Byte Bandit. Contra acestora s-au creat doi antiviruși puternici :

• *VIRUSX*. Acest antivirus este garantat contra virusului standard ce atacă blocul *boot*.

Detectarea virusului decurge astfel :

— conectați calculatorul înainte de a pune dischetă în DFØ ;

— formați o dischetă virgină ;

— „stingeți" calculatorul și apoi relansați discheta formatată în citire ;

— dacă sosește o *boot-are*, aceasta arată că discheta este virusată.

Distrugerea virusului se poate realiza astfel :

— procurați o dischetă sănătoasă 100% ;

— „aprindeți" calculatorul înainte de a pune discheta în DFØ ;

— deschideți o fereastră CLI, tastați comanda

INSTALL ?

și validați prin Return ;

— amplasați discheta sănătoasă, „peste" cea contaminată ;

— tastați

DFØ :

și validați prin Return.

Comanda INSTALL rescrie sectoarele de început ale dischetelor. Ea este foarte eficace contra virușilor instalați în sectorul *boot* al dischetei. Dar dacă virusul operează diferit ? Dacă virusul nu se găsește în *boot* ?

În acest caz, trebuie utilizat un alt antivirus.

Un nou tip de virus care nu modifică *boot*-ul și operează mult mai discret este IRQ. El atacă CLI. O simptomă a prezenței sale este următoarea : cînd deschideți o fereastră, titlul acesteia cuprinde și mesajul

New Virus by the IRQ-Team V 41.0\B

Acest IRQ, care este puțin cunoscut, declanșează mesaje aleatoare. El poate, de exemplu, cere scoaterea protecției la scriere pe dischetă și deci poate amenda fișierele-rezultat, expandîndu-le.

VIRUSCAN

Programul scanează dischete sau întregul sistem și identifică orice existență a infecțiilor cu viruși. El va indica fișierele specifice sau ariile sistemului care sînt infectate și va identifica specia de virus care a cauzat infecția. Ștergerea poate fi făcută automat, folosind opțiunea SCAN/D. Dacă infecția este larg răspîndită, sînt disposibile utilitarele automate de dezinfecție care pot șterge segmentul infectat al fișierelor și pot repara și restaura programele infectate.

Versiunea SCAN 6.0V71 identifică 260 de specii de viruși și numeroase subvarietăți ale acestora. Aceștia realizează peste 95% din toate infecțiile de PC raportate.

Toți virușii cunoscuți infectează una din ariile următoare : tabela de partiție a HD, sectorul de *boot* ; unul sau mai multe fișiere executabile (programe ale sistemului de operare, drivere ale dispozitivelor sistemului, fișiere .COM, fișiere .EXE, fișiere de reacoperire sau orice alt fișier care poate fi încărcat în memorie și executat). *VIRUSCAN* poate verifica întregul sistem, o dischetă individuală, un subdirector sau un fișier individual.

Mod de lucru. Pentru a lucra cu VIRUSCAN se va tasta :

SCAN d1 : d2 ... : dn : [/M /D /A /E [EXTENSION LIST] /nomem/m]

unde :

d1, d2, ... — indică driverele ce vor fi scanate ;
/D — suprascriere și ștergere fișiere infectate ;
/M — scanează memoria pentru toți virușii ;
/A — scanează toate fișierele ;
/E — scanează reacoperirile listate ;
/nomem — omite scanarea memoriei
/m — scanează multe FD.

VIRUSCAN va verifica fiecare arie sau fișier de pe dispozitivul desemnat, care poate fi gazdă pentru un virus. Dacă este găsit un virus, se va afișa numele virusului și numele fișierului/ariei infectate.

● Dacă este selectată opțiunea /D, programul SCAN se va opri după afișarea fiecărui fișier infectat și va întreba dacă se dorește ștergerea fișierului. Dacă se răspunde cu Y (yes = da), atunci fișierul este suprascris cu codul hexa C3 (instrucțiunea Return) și apoi șters. Utilitarele M-DISK vor distruge virușii sectorului de boot sau ai tabelei de partiții.

● Dacă este aleasă opțiunea /M, SCAN va căuta în primii 640 K de memorie toți virușii cunoscuți rezidenți în memorie. Alegerea acestei opțiuni poate cauza alarme false, dacă se rulează SCAN împreună cu alt utilitar de detecție a virușilor. De asemenea, se adaugă 12 secunde pînă la un minut la timpul de scanare. Dacă nu este aleasă operațiunea /M, programul va controla, totuși, memoria de viruși Dark Avenger. În cazul cînd Dark Avenger nu este găsit, SCAN va afișa un mesaj de avertizare cu instrucțiuni pentru a dezactiva și a reîncărca dintr-un *floppy* curat.

● Opțiunea /E este folosită pentru a scana fișierele de reacoperire specificate. SCAN nu își va îndeplini îndatoririle cu OVL, OVG, OV1, OV2, OVR, SYS, BIN, PIF. Programul SCAN va cerceta aceste fișiere de existența lui Jerusalem, Vacsina, Fu Manchu și Dark Avenger (viruși care sînt capabili de infectarea acestui tip de

fișiere). Dacă folosim o aplicație cu extensii de reacoperire, altele decît cele implicite, atunci vom specifica numele extensiilor (pînă la 3) folosind opțiunea /E. De exemplu,

SCAN C : /E .ABC .XYZ .123

Virușii care infectează structura de reacoperire vor infecta întotdeauna și fișierele originale .COM, .EXE, .BIN, .SYS care cheamă reacoperirea. Deci, virusul va fi întotdeauna descoperit dacă structura de reacoperire este scanată sau nu. Pentru a scăpa de virus, trebuie să-l identificăm și să-l ștergem. Dacă nu se știe faptul că aplicația folosește fișiere de reacoperire și SCAN a descoperit unul dintre virușii capabili să infecteze *overlay*, atunci se va folosi opțiunea /A pentru a cerceta toate fișierele.

● Opțiunea / A necesită o cantitate substanțială de timp pentru a completa scanarea. O vom folosi doar după ce a fost descoperită o infecție .COM sau .EXE de către VIRUSCAN.

Programul poate de asemenea scana directori individuali sau fișiere individuale. De exemplu, comanda

SCAN C : \ DIRECT \ PROGRAM.EXE

va scana fișierul PROGRAM.EXE în subdirectorul DIRECT.

Pentru fiecare 100 fișiere ale dispozitivului desemnat, *VIRUSCAN* consumă aproximativ trei minute timp de rulare. Codurile de ieșire ale programului SCAN sînt :

0 — pentru terminarea normală și viruși negăsiți ;
1 — pentru unul sau mai mulți viruși depistați ;
2 — pentru terminarea anormală (eroare).

VIRUSCAN lucrează doar pe PC-uri individuale. Dacă sîntem într-o rețea locală atunci va trebui să rulăm NETSCAN.

Versiuni. Programul posedă mai multe versiuni, fiecare avînd un rol bine definit.

● Versiunea 54 fixează o defecțiune din versiunea 53 care a cauzat o alarmă falsă pentru virusul 4096 în cîteva situații.

• Versiunea 53 nu permite specificarea mai multor dispozitive în linia de comandă. Fișierele cu extensii PRG au fost adăugate la lista de extensii *overlay* pentru scanarea automată. Versiunea detectează noii viruși : Oropax, 4096, Virus-90 și Chaos. Ea este mai rapidă și asigură compatibilitatea cu noul dezinfector CLEAN-UP.

• Versiunea 52 detectează, în plus, viruții Devil's Dance (rezident în memorie și amestecă toate datele pe driverul hard) și Aids Information Trojan (blochează întregul driver C cînd este activat după 90 de rebuturi).

• Versiunea 51 asigură două noi opțiuni de scanare :
/ nomem — omite scanarea memoriei ;
/ m — scanează multe FD-uri.
Această versiune detectează și viruții Payday, Datacrime II-B și Amstrad. Virusul Payday, care este o variantă a lui Jerusalem, activează în fiecare vineri, exceptînd ziua de 13.

Datacrime II-B este o versiune modificată a lui Datacrime II care folosește o tehnică diferită de decriptare pentru a evita detecția. Virusul Amstrad este un infectator COM care conține un anunț pentru calculatorul Amstrad. Dacă Payday și Datacrime II-B sînt distrugători, virusul Amstrad se pare că este benign.

• Versiunea 50 detectează virusul Holland Girl ce infectează fișiere COM, pe care le mărește cu 1 332 bytes.

• Versiunea 49 descoperă existența următorilor viruși : Donothing (care infectează fișierele .COM, dar nu face altă stricăciune și nu afectează sistemul într-un mod observabil), Lisbon (infectează fișierele .COM, mărește dimensiunea programelor infectate cu 648 bytes ; el distruge pînă la opt programe prin suprascriere a AIDS peste primii cinci bytes ai programului infectat), Sunday Virus (activează numai duminica afișînd mesajul „Azi este duminică, de ce muncești așa din greu ?").

• Versiunea 48 identifică virusul TYPO.COM care afectează datele trimise la portul paralel.

• Versiunea 47 identifică virusul DBASE care este un infectator COM ce corupe datele în fișierele DBF.

• Versiunea 46 include un test pentru virusul Ghost care atacă fișierele .COM din sectoarele de *boot* ale HD și FD. Fișierul în cauză își mărește dimensiunea cu 2 351 bytes și înlocuiește sectorul de *boot* cu un virus de *boot* similar cu Ping Pong. De asemenea, versiunea include :
— o opțiune pentru a șterge fișierele infectate prin suprascrierea acestora și apoi ștergerea lor ;
— scanarea memoriei este opțională pentru cei mai mulți viruși rezidenți în memorie ; totuși, verificarea memoriei este forțată doar pentru virusul Dark Avenger ;
— șirurile de scanare au fost schimbate încă o dată pentru a evita alarme false.

• Versiunea 45 detectează noua versiune a lui Jerusalem.

• Versiunea 43/44 identifică virusul Alabama care infectează fișiere EXE, mărindu-le dimensiunea cu 1 560 bytes, manipulează FAT și deplasează nume de fișiere. De asemenea, versiunea verifică prezența lui Dark Avenger în memorie.

Adăugirile versiunii 43/44 sînt : include căutări în fișiere .EXE pentru Datacrime II ; identifică virusul Pakistani Brain cît timp el este activ în memorie ; semnal audio dacă sînt găsiți viruși ; accelerează căutările pentru mulți subdirectori.

• Versiunea 42 include un identificator pentru virusul Yankee Doodle (vezi SCANRES).

• Versiunea 41 este o replică la emiterea de către IBM a propriului produs de scanare. Ea face verificări pentru 28 de viruși, din care doi nu erau cunoscuți de *VIRUSCAN*.

SCANRES

Versiunea 42 a lui *VIRUSCAN* include un identificator pentru virusul Yanke Doodle. Virusul cîntă melodia „Yanke Doodle Dandy" la difuzorul sistemului după 17 ore de la încărcarea programului infectat. Fișierele infectate .COM și .EXE cresc cu peste 2899 bytes.

Infectare. Viruții detectați cu *SCANRES* infectează una din ariile următoare : tabela partiției discului hard, sectorul de *boot*, una sau mai multe fișiere executabile

din interiorul sistemului. Fișierele executabile pot fi : programe ale sistemului de operare, drivere ale dispozitivelor sistemului, fișiere .COM, fișiere .EXE, fișiere de reacoperire și orice alt fișier care poate fi încărcat în memorie și executat.

Identificare și prevenire. SCANRES, care este versiunea rezidentă în memorie a lui VIRUSCAN, previne intrarea virușilor în sistem. El monitorizează și scanează programe încărcate și previne execuția programelor infectate. De asemenea, el scanează sectorul de boot, tabela de partiții, fișiere, întrerupătorul Command și el însuși, cînd este executat prima dată.

Modulul SCANRES .EXE identifică numele virusului care a cauzat infectarea și aria sau fișierul care a devenit infectat. După ce este încărcat el rămîne activ în sistem tot timpul.

Versiunea SCANRES 0.9.V42 identifică 42 viruși distincți, sau subspecii ale acestora: Yanke Doodle, 2930, Ashar, Aids/VGACGA, Disk Killer/Ogre, 1536/Zero Bug, Mixl, Dark Avenger, 3551/Syslock, Vacsina, Ohio, Typo, Swap/Israeli Boot, 1514/Datacrime II, Icelandic-II/System Virus, Pentagon, 3066/Traceback, 1168/Datacrime-B, Icelandic, Saratoga, 405, 1704 Format, Fumanchu, 1280/Datacrime, 1701/Cascade, 1704/Cascade-B, Stoned/Marijuana, 1704/Cascade/Falling Letters, Ping Pong-B/Falling Letters Boot, Den Zuk, Ping Pong/Italian/Bouncing Dot, Viena-B, Lehigh, Vienna/648/DOS-68, Jerusalem-B, Yale/Alameda, Friday 13 th COM Virus, Jerusalem/1813, Suriv03, Suriv02, Suriv01, Pakistani Brain. După cum se observă, unii viruși au diferite versiuni rezultate din modificări ale versiunii originale. De exemplu, DEN ZUK are două versiuni separate. *SCANRES* scanează aproape identic toți virușii din aceeași grupare (indiferent de versiune). Dacă acest lucru nu este posibil, atunci *SCANRES* va face diferențierile de rigoare.

SCANRES devine activ de fiecare dată cînd sistemul este activat sau re-*boot*-at. El va controla de viruși ariile critice ale sistemului, inclusiv pe sine ; se vor monitoriza toate încărcările de program. Dacă programele sînt încărcate, *SCANRES* le va scana căutînd viruși. În cazul găsirii unuia, se va afișa un mesaj de avertizare și va numi infecția. Programul infectat va fi întrerupt.

Distrugerea virusului. Ce faceți dacă este găsit un virus ? În acest caz trebuie să acționați în funcție de tipul infecției, și să parcurgeți anumiți pași. Astfel,

- pentru infecții ale sectorului de *boot* executați următoarele : dezactivați sistemul : activați-l și *boot*-ați dintr-un FD neinfectat, protejat la scriere ; executați comanda DOS SYS pentru a încerca o suprascriere a sectorului de boot. Schema aceasta funcționează în multe cazuri. Dacă ea dă greș, atunci salvați toate fișierele de date și realizați o formatare de nivel scăzut a discului.
- pentru infecții ale fișierelor executabile acționați astfel : ștergeți toate fișierele infectate și înlocuiți-le cu cele din dischetele originale.
- pentru infecții ale tabelei de partiții se recomandă să formatați mediile la un nivel scăzut.

Mod de operare. Mai întîi se plasează SCANRES pe un FD protejat la scriere. Aceasta va asigura o copie validă în eventualitatea în care programul devine infectat.

Pentru a instala *SCANRES* puneți linia ce conține numai cuvîntul *SCANRES* ca primă linie în fișierul AUTOEXEC.BAT și apoi copiați fișierul *SCANRES* în directorul *root* din driverul hard *boot*-abil (de obicei C :).

SCANRES necesită 17 Kb memorie. El va adăuga, în medie, opt secunde la fiecare încărcare de program. După ce un program a fost încărcat și a început execuția, *SCANRES* nu va micșora performanța sau viteza sistemului în nici un fel.

VALIDATE

Versiunea o.2 a lui *VALIDATE* este un program de *autentificare* a fișierelor. El folosește două metode diferite pentru a genera verificări asupra fișierelor desemnate și afișează rezultatele. Verificările asigură un nivel ridicat de securitate. Rezultatele pot fi apoi comparate cu datele de validare publicate de autor sau obținute de orice sursă sigură de informații.

Rulînd programul *VALIDATE*, care operează asupra programului utilizator, vom obține numerele de autentificare pe care le comparăm cu cele din fișierul bazei de

date. Dacă acestea se potrivesc atunci programul utilizator nu a fost falsificat.

Pentru a lansa în execuție programul VALIDATE vom tasta linia :

VALIDATE d : \ path \ filename.ext

iar programul va afișa următoarea informație de validare :

Size : (număr de bytes)
Date : (data creării fișierelor)
File Authentification :
Check Method 1 — (patru caractere)
Check Method 2 — (patru caractere)

CLEAN-UP

Programul *CLEAN-UP* dezinfectează și repară programe infectate. Unele versiuni ale programului sînt împachetate cu *VALIDATE* care va autentifica integritatea programului executabil CLEAN .EXE, rezultatele validării fiind :

Size :
Date : 01-09-1990
File Authentification :
Check Method 1 — 4345
Check Method 2 — 025 D

Dezinfectarea. Programul *CLEAN-UP* distruge și șterge viruși, dar în cele mai multe cazuri repară fișiere infectate, reconstruiește programe deteriorate și pune sistemul pe o operare normală.

CLEAN-UP cercetează întregul sistem, căutînd virusul pe care dorim să-l ștergem. O dată ce virusul este găsit, el este izolat și șters, iar pentru virușii obișnuiți fișierul infectat este reparat. În cazul infectării cu un virus mai puțin cunoscut, care nu poate fi separat din fișiere, fișierul infectat este șters total din disc după ce este afișat un mesaj de avertizare.

Virușii obișnuiți pe care *CLEAN-UP* poate să-i șteargă cu succes, să repare și să restaureze programele avariate sînt : Alabama, Jerusalem A, Jerusalem B, Jerusalem E, Ping Pong, Ping Pong B, Suriv-03,1704, Stoned, Payday, Disk Killer, Alameda, Dark Avenger, Ashar, 1701, Pakistani Brain.

Înainte de a rula *CLEAN-UP* verificăm infecția cu ajutorul programului *VIRUSCAN* (SCAN.EXE). *SCAN* va identifica specia și subspecia virusului și va afișa identificatorul ID care va fi folosit ca intrare în CLEAN-UP. Identificatorul virusului este afișat în interiorul unui set de paranteze[]. De exemplu, identificatorul virusului Disk Killer este [Killer].

De asemenea, tot înainte de a începe dezinfectarea vom dezactiva calculatorul infectat, îl vom reîncărca dintr-o dischetă sistem curată protejată la scriere. Acest pas este foarte important deoarece va muta virusul din control în memorie.

Pentru a rula programul dezinfectator vom testa comanda :

CLEAN d1 :d2 :...: dn : ⟨id virus⟩ /a/opțiune

unde

d1, d2,... desemnează decriptori de dispozitive care trebuie curățate ; la o singură comandă pot fi curățate maxim 10 dispozitive ;

[id virus] — identificatorul virusului ;

/a — opțiune de verificare a fișierelor ;

/opțiune — opțiune ce permite curățarea de mai multe FD.

Exemple :

CLEAN C : D : [Jeru] va curăța virusul Jerusalem din discurșurile C și D ;

CLEAN C : \TEMP [Dav] /a va curăța Dark Avenger din C : \TEMP și va căuta toate extensiile de fișier pentru virus

Dacă un fișier a fost infectat de mai multe ori de un virus, CLEAN va afișa numele fișierului și mesajul de „succes" pentru fiecare apariție a infecției.

După rularea programului *CLEAN-UP*, rulați iar *SCAN* cu opțiune /a pentru a vă asigura că toate remanențele virusului au fost șterse.

Identificatorii. Pentru fiecare dintre virușii cunoscuți există cîte un identificator. Astfel, în lista următoare

după fiecare virus, urmează între paranteze numele identificatorului :

PingPong/Bouncing Dot [Ping], Ping-Pong-B/Cascade Boot [Ping], Suriv01 [April], Suriv03 /Jerusalem- E [Jeru], Suriv 02 [Jeru-D], Jerusalem-A1813 [Jeru], Jerusalem-B [Jeru], New Jerusalem [Jeru], Friday 13th COM virus [13], Stoned/Marijuana [Stoned], Vienna-B [Vienna-B], Vienna/DOS-62 [Vienna], Yale/Alameda [Alameda], Lehigh [Lehigh], Alabama [Alabama], 2930 [2930], 1704 Format [170X] 1704/Cascade-B [170X], 1704 Cascade [170X] 405 [405], 1701/Cascade [170X], 3066/Tracebak [3066], Fu Manchu/2086. [Fu], Icelandic [Ice], Icelandic-IISystem [Ice-2], Chaos [Chaos], Virus-90 [90], Pakistani Brain [Brain], Devil's Dance [Dance], Datacrime II-B [Crime-2B], Datacrime II [Crime-2], Datacrime-B [Crime-B], 1280/Detacrime [Crime], Saratoga [Toga], Vacsina [Vacs], Mixl [Mixl], 3551/Syslock [Syslock], Disk Killer/Ogre [Killer], Ashar [Brain], Yankee Doodle [Doodle], Typo COM virus [Typo], Sunday Virus [Sunday], Dark Avenger [Dav], 1536/Zero Bug [Zero], AIDS/Taunt [Taunt], Oropax [Oro], 4096 [4096], AIDS Trojan [AIDS], Amstrad [Amst], Holland Girl [Holland], Do-Nothing virus [Nothing], Lisbon virus [Lisb], DBASE virus [Dbase], Ghost COM version [Ghost-C].

Așa cum afirmam mai înainte, lista medicilor informatici rămîne deschisă... NORTON ANTIVIRUS (NAV) este unul dintre cele mai recente, complete și evoluate programe antivirus disponibile pentru IBM-PC. NAV este alcătuit din două module : VIRUS INTERCEPT și VIRUS CLINIC. Primul este un program rezident lansat la pornirea calculatorului (o linie specială inclusă în CONFIG. SYS) ; el semnalizează apariția unui virus evitînd astfel propagarea lui. Al doilea este elementul de detecție și distrugere a virușilor. NAV poate fi actualizat incluzînd în Fișierul NAV.DEF semnăturile noilor viruși.

8.

VIRUȘI VECHI, VIRUȘI NOI

Spre deosebire de SIDA, un virus informatic nu vrea să ne omoare, dar ne poate provoca mult rău atunci cînd distruge programele și datele. Din nefericire, virușii informatici au devenit ceva obișnuit în lumea PC-urilor, fiind practic o realitate a vieții. Important este să ne „păzim". Pentru aceasta este nevoie să ne cunoaștem adversarul.

Asociația industrială a Virușilor Informatici (*C*omputer *V*irus Industry *A*socciation — CVIA) din SUA face următoarea clasificare a virușilor :

— CLASA 1 — viruși care infectează orice program ;

— CLASA 2 — viruși care infectează fișierele-sistem (extensia .SYS) ;

— CLASA 3 — viruși care infectează sectorul de autoîncărcare sau tabela de partiții.

Deoarece virușii care infectează fișierele-program (extensiile .COM sau .EXE) sînt adesea astfel constituiți încît să atace și fișiere ale sistemului (de exemplu, COMMAND.COM și chiar fișiere cu extensia .SYS), convenim să considerăm împreună primele două clase ca viruși care infectează fișiere. În continuare se prezintă manifestările și „pagubele" unor viruși mai vechi sau mai noi.

8.1. VIRUȘI CARE INFECTEAZĂ FIȘIERE

Virușii din această clasă se fixează de regulă pe fișierele cu extensia .EXE sau .COM. În momentul în care programul infectat este rulat prin diverse căi, virusul se

activează, instalîndu-se în memoria RAM unde rămîne rezident pînă la o nouă încărcare a sistemului. Orice program rulat după aceasta poate fi de asemenea infectat. În funcție de specificul virusului, infecția poate fi selectivă. Dimensiunea programului infectat crește exact cu mărimea virusului. Întrucît această dimensiune devine astfel o cale de identificare a virusului, majoritatea virușilor care infectează fișiere au și un al doilea nume dat de mărimea virusului („VIENNA" sau „648" ; „ALABAMA" sau „1 560" etc.).

De regulă, virusul infectează fișiere „curate", dar se cunosc și viruși care infectează repetat același fișier (exemplu : Jerusalem).

Virusul FRIDAY 13 th

Acest virus de tip Jerusalem B, realizat în Israel, a fost descoperit de CARMEL Software Engineering în iulie 1987. Friday 13th are o mărime de 1 808/1 813 octeți, incluzînd și ultimii 5 octeți care sînt semnul infecțării. Virusul infectează fișierele de tip .EXE și .COM. Odată instalat într-un astfel de fișier, sarcina lui este de a se întreține și a se multiplica.

Cînd este executat un program virusat, Friday 13 th, folosindu-se de funcția DOS 31H prin întreruperea 21H, se „ascunde" în memorie, rămîne rezident (TSR), ocupînd între 1,8 și 2,2 ko. Rularea unui program .COM sau .EXE va duce la infectarea acestuia. Dimensiunea fișierelor .COM este mărită cu 1 813 bytes, iar a fișierelor EXE cu 1 808 bytes. Fișierele .COM sînt infectate o singură dată, în timp ce fișierele .EXE vor fi reinfectate la fiecare rulare. Ele devin din ce în ce mai mari și din ce în ce mai lente.

Friday 13th își face simțită prezența prin apariția unor „cutii" negre pe ecran și, ce este mai grav, prin ștergerea tuturor fișierelor care sînt executate în fiecare vineri în care este 13 (Vinerea neagră).

Virușii APRIL 1st (tipul A și B)

Cunoscut și sub numele de virus TSR sau SURIV 1, April 1st infectează numai fișierele .COM. A fost descoperit în Israel în iulie 1987 și, asemănător cu Friday

13th, rămîne rezident, dar nu infectează fișierele .EXE, avînd o dimensiune mai mică (897 octeți).

Nu este un virus foarte răspîndit, fiind foarte ușor de identificat, întrucît atunci cînd infectează un nou fișier, pe ecran apare scris „YOU HAVE A VIRUS" iar pe data de 1 aprilie va afișa :

„HA HA HA IT'S APRIL 1ST — YOU HAVE A VIRUS".

Virușii SURIV 2&3 (April 1 st D)

Spre deosebire de „April 1st" (tipul A și B), virușii SURIV 2, SURIV 3 sau Jerusalem D, care sînt tot viruși de tip „April 1st", infectează numai fișierele .EXE, au o dimensiune mai mare (1 488 octeți), nu rămîn rezidenți, dar vor acționa utilizînd blocurile de control al memorie (MCB — *M*emory *C*ontrol *B*locks). SURIV 2 diferă de SURIV 3 prin mici modificări în primii 16 octeți, restul codului fiind identic.

Virusul TYPO

Acest virus infectează numai fișierele .COM. El are o mărime de 867 octeți și, folosind întreruperea 21H, funcția 31H rămîne rezident.

Cînd un fișier infectat este executat, virusul va infecta toate fișierele de tip .COM pe care le va găsi neinfectate în directorul curent.

„Typo" schimbă în mod aleator valorile actuale ale tastelor cu altele. De exemplu, în loc de „0" va tipări „x", în loc de „\" va tipări „m" etc.

Virusul 640k

Deoarece nu are nici un scop distructiv, virusului 640 k i se mai spune și „virusul stupid" sau „virusul care nu face nimic". Dar ceva tot face : infectează fișierele de tip .COM, mărindu-le cu dimensiunea lui (183 octeți). El va rămîne în memorie, autocopiindu-se la adresa 9 800:0000. Așa se explică prezența lui numai pe sistemele cu 640 KB.

Atunci cînd este executat un fișier infectat, virusul va căuta printre toate fișierele de tip .COM ale directorului curent un fișier neinfectat și-l va infecta.

Virusul MIXER1

MIXER 1, descoperit în august 1989, în Israel, are două versiuni : versiunea A (1 615 octeți) și versiunea B (1 635 octeți). El infectează doar fișierele cu extensia .EXE și este rezident. Dar spre deosebire de alți viruși rezidenți nu utilizează funcția 31h cu întreruperea 21h pentru a rămîne rezident. Se manifestă prin mixarea (amestecarea) datelor care sînt transferate spre porturile seriale și paralele.

Mărimea programelor infectate crește cu 1 618/1 635 octeți (+ 15 octeți pentru alinierea paragrafelor). La sfîrșitul infectării fișierului poate fi găsit șirul MIX1.

La execuția programului CHKDSK dacă virusul este în memorie, valoarea indicată pentru memorie va fi cu 2KB mai mică decît valoarea reală.

Virusul ALABAMA

Cunoscut și sub numele de 1 560, nume inspirat de mărimea programului (1 560 octeți), este unul din cei mai mici și mai distructivi viruși care infectează fișiere. A fost descoperit în septembrie 1989. Asemănător virusului MIXER1, atacă fișierele .EXE și nu folosește funcția 31h cu întreruperea 21h pentru a rămîne rezident. Dar spre deosebire de MIXER1, ALABAMA se autoîncarcă sub cea mai înaltă locație de memorie raportată de DOS și nu micșorează valoarea afișată de BIOS sau DOS pentru memorie, fapt ce ridică mari probleme.

Virusul „fură" întreruperea 09h și verifică tastatura doar pentru RESET și combinația de taste ALT + CTRL + DEL (pornirea la cald) și va utiliza ciclurile de intrare (IN) și ieșire (OUT). Cînd depistează cererea de reîncărcare a sistemului, utilizînd rutina de autoîncărcare va reîncărca sistemul dar el va rămîne ascuns în memorie.

Virusul va controla ora sistemului la orice apel al întreruperilor 9h sau 12h, iar după o oră de la activare va afișa mesajul :

SOFTWARE COPIES PROHIBITED BY INTERNATIONAL LAW..*
Box 1055 Tuscambia ALABAMA USA.

El nu infectează direct programul activ, dar este unul dintre cei mai pricepuți viruși în infectarea sistemului. Mai întîi va căuta în directorul curent dacă există fișiere neinfectate și dacă găsește, bineînțeles că va infecta unul. Dacă nu găsește nici un fișier neinfectat îl infectează pe cel aflat în execuție. Uneori, cînd virusul găsește un fișier neinfectat, în loc să-l infecteze îl va schimba cu fișierul aflat în execuție. Schimbarea o face fără renumirea fișierului ceea ce generează dezinformarea utilizatorului (schimbă între ele de exemplu DISCOPY cu FORMAT ; utilizatorul știe că face DISCOPY și de fapt își formează discul).

Virusul 100 YEARS

Este unul dintre cei mai „deștepți" viruși. A fost descoperit în octombrie 1989, în Israel. Are 4 096 octeți (de aceea este denumit și „4K" sau „4 096") și infectează fișierele cu extensia .EXE și .COM (inclusiv COMMAND.COM-ul). Nu folosește INT 21h cu funcția 31h pentru a rămîne rezident. Se autoîncarcă la cea mai mare locație de memorie raportată de DOS, dar nu va modifica mărimea actuală a memoriei raportată de BIOS. DOS-ul va raporta cu 6K mai puțin decît BIOS (CHKDSK).

Mărimea fișierelor infectate crește cu 4K, iar în director, anul corespunzător fișierului infectat este mărit cu 100 (de unde și numele). Utilizatorul nu va sesiza prea repede fișierele infectate, întrucît la afișarea datei sînt furnizate numai ultimele două cifre ale anului, iar la execuția comenzii DIR „100 Years" are grijă să afișeze mărimea inițială a fișierului (cea actuală micșorată cu 4 Ko).

Furtul de întrerupere este foarte scurt și virusul nu poate fi detectat de utilitare (PCTOOLS, SMAP, SNOOP...). „Păcălește" de asemenea și utilitarele de atenționare care verifică întreruperile pentru acțiuni „ilegale" de virusare.

Virusul VIENNA

A fost descoperit în Israel, de CARMEL Software Engineering, în noiembrie 1989 și este cunoscut și sub numele de „DOS 62 seconds" și „648".

Spre deosebire de alți viruși, VIENNA nu se încarcă singur în memorie. Cînd un fișier infectat se află în execuție, virusul va căuta în PATH-ul curent un fișier neinfectat și-l va infecta. Infectează numai fișiere de tip .COM (inclusiv COMMAND.COM) cărora le mărește dimensiunea cu 648 octeți.

Acest virus distruge, în mod aleator, fișierele înlocuindu-le primii cinci octeți, de fiecare dată cînd rezultatul funcției SI-logic (instrucțiunea de asamblare AND) între timpul sistem exprimat în secunde și cifra 7 este zero.

Virusul SYLVIA

De ce SYLVIA ? Pentru că virusul ne recomandă să scriem o ilustrată frumoasă unei fete pe nume Sylvia, a cărei adresă o și dă (de aici și al doilea nume sub care este cunoscut : „Netherlands girl"). Ni se promite în schimb un antivirus pentru „Sylvia"...

Virusul are 1 332/1 331 octeți și infectează numai fișierele .COM, mai puțin COMMAND.COM.

Se autoîncarcă în memorie folosind blocul de control al memoriei (MCB). După activare driverul curent devine C :. Infectează fișierele din directorul rădăcină, din directorul curent, precum și fișierele în curs de execuție. O parte din fișiere le distruge rămînînd în ele numai programul virus (1 332 octeți).

Virușii DATACRIME

Cunoscuți și sub numele de „Columbus day", acești viruși afișează pe ecran, în ziua de 12 octombrie, următorul mesaj :

1 MARCH 1989, DATACRIME VIRUS

Partea neplăcută urmează după afișarea mesajului, și anume formatarea discului hard. E drept că formează o singură pistă, dar nu oricare, ci prima. Așa că...

Variantele în care pot fi întîlniți sînt :

a) DataCrime A — are 1 168 octeți, infectează doar fișierele .COM, încărcîndu-se în memorie utilizînd MCB. Este un virus neplăcut, care a produs multora necazuri.

b) DataCrime B — asemănător cu DataCrime A, fapt pentru care este și confundat cu acesta de unele programe de antivirusare. Se mai numește și 1 280 deoarece are 1 280 octeți.

c) DataCrime C — infectează fișierele .EXE și .COM ; cînd un fișier infectat este executat, virusul caută fișierele neinfectate în toate directoarele driverelor A,B și C și-l infectează pe primul găsit. În formă criptată virusul se lipește de orice fișier executabil care are mai puțin de 60 K (inclusiv COMMAND.COM). Are 1 514 octeți.

Virusul DARK AVENGER

Încărcat în memorie prin MCB, virusul infectează fișierele .EXE și .COM. Scopul său este să scrie în sectorul de autoîncărcare și să „aranjeze" tabela de alocare a fișierelor. Dark Avenger conține mesajul :

Eddi lives somewhere in time

Virusul FU-MANCHU A

Iată un virus nostim. Asemănător cu Dark Avenger, se încarcă în memorie prin MCB și infectează fișierele .EXE și .COM. Blochează întreruperile 21h, 9h, iar din august 1989 și întreruperea 16h. Virusul are 2 086 octeți. Fiind „stăpîn" pe întreruperile cu tastatura, cînd se face încărcarea la cald a sistemului (ALT + CTRL + DEL) FU-Manchu va afișa pe ecran mesajul :

The World will hear from me again ! ! !

Mesajul va fi șters dacă se răspunde cu „fuck". Prezentăm mai jos alte cîteva răspunsuri posibile și continuarea lor :

Răspuns	*Continuare*
'Waldheim"	'is a nazi'
'thacher'	'is a cunt'
'reagen'	'is an arshole'
'FUManchu'	'virus date :' etc....

Virusul ICELANDIC A

Sistemul de infectare al virusului Icelandic A este foarte asemănător cu cel al virusului MIXER și cu siguranță a fost sursa de inspirație pentru acesta din urmă. „Bătrînul" Icelandic A infectează fișierele cu extensia .EXE și rămîne rezident, autoîncărcîndu-se în partea superioară a memoriei. Valoarea memoriei raportate de DOS este micșorată cu 2K.

Prima versiune a virusului (642 octeți) infectează fiecare al 10-lea fișier care este executat, prin asta explicîndu-se și slaba sa răspîndire. A doua versiune (661 octeți) infectează orice fișier .EXE, aceasta fiind singura diferență față de prima versiune.

Virusul PRETORIA

Infectează numai fișierele .COM, inclusiv COMMAND.COM și adesea IO.SYS. Dacă este infectat fișierul sistem IO.SYS, se impune executarea comenzii SYS pentru reutilizarea discului ca disc sistem. Virusul PRETORIA are 879 octeți și nu se autoîncarcă în memorie. La 16 iunie, într-un sistem infectat, toate fișierele din directorul rădăcină vor fi renumite ZAPPED.

Virusul CHRISTMAS

Un virus drăguț : în perioada 24 decembrie-1 ianuarie desenează pe ecran un pom de crăciun. Este cunoscut și sub numele XA1. Ocupă 1 539 octeți, nu se autoîncarcă în memorie și infectează numai fișierele .COM.

Virusul MACHOSOFT

Un alt virus care infectează fișierele .COM (inclusiv COMMAND COM) este Machosoft. Infectează numai fișierele din driverul curent, fișierele din celelalte drivere fiind în siguranță.

Machosoft are 3 551 octeți (+0—15 pentru aliniere) și nu se autoîncarcă în memorie. De fiecare dată cînd este executat un fișier infectat, virusul caută în driverul curent fișierele „sănătoase" și infectează, în mod aleator, un fișier .COM sănătos. Virusul creează un fișier cu atributele ascunse și cu acces numai în citire (*hidden/read only*) botezat IBMIONET.SYS. Cînd infectează

un fișier virusul citește 20 de sectoare (folosind întreruperea 25h) și memorează numărul ultimului sector în fișier. Caută în cele 20 de sectoare șirul MICROSOFT. De fiecare dată cînd îl găsește îl înlocuiește cu MACHOSOFT, apoi rescrie cele 20 de sectoare.

Virusul OROPAX

Oropax infectează fișierele .COM, mai puțin fișierul COMMAND.COM. Un fișier infectat va avea întotdeauna dimensiunea divizibilă cu 51. De aceea virusul Oropax va infecta numai fișierele a căror dimensiune nu este divizibilă cu 51 iar mărimea virusului este variabilă (2 756-2 806 octeți). Rămîne rezident în memorie și infectează fișierele directorului curent cînd întreruperea 21h este folosită pentru apelul funcțiilor : 13h — ștergerea unui fișier utilizînd FCB (comanda DEL din MS-DOS) ; 16h — crearea unui fișier utilizînd FCB ; 17h — redenumirea unui fișier, utilizînd FCB (comanda REN) ; 39h — crearea unui nou director (MKDIR) ; 3Ah — ștergerea unui director (RMDIR) ; 3Ch — crearea unui fișier (CREAT) ; 3Dh — deschiderea unui fișier, utilizînd un identificator logic ; 41h — ștergerea unui fișier (UNLINK, respectiv DEL din MS-DOS) ; 43h — obținerea (poziționarea) atributelor fișierului (CHMOD) ; 46h — redirectarea unui identificator logic (FORCDUP).

În cîteva minute de la activare, în mod aleator („decizia" aparține virusului), virusul va „cînta" trei melodii diferite, cu o pauză de 7 minute între ele. Explicabilă deci a doua denumire a virusului : MUSIC.

Virusul TUESDAY 1st

Este aproape la fel cu virusul Friday 13th. Două diferențe există între ele : a) virusul este activat cînd prima zi a lunii este marți ; b) poate fi identificat întrucît schimbă MSDOS în MsDns.

Virusul SATURDAY 14th

Cunoscut și sub numele de Durban, virusul Saturday 14th are 681 octeți și infectează fișierele .EXE și .COM (exclusiv COMMAND.COM-ul). În a 14-a zi din lună, dacă este joi, virusul distruge discul scriind în primele 100 de sectoare.

Virusul SUNDAY

Virusul rămîne rezident utilizînd funcția 31h prin întreruperea 21h. Are 1 636 octeți și infectează fișierele EXE și .COM ce vor fi executate după instalarea sa. În fiecare duminică virusul va afișa pe ecran următorul mesaj :

„Today is SunDay ;
Why de you work so hard ?
All work and no play make you a dull boy !
Come on ! Let's go out and have some fun !

Acest virus poate corupe și tabela de alocare a fișierelor „furînd" întreruperea 8h.

Virusul ZEROBUG

Acest virus este cunoscut și sub denumirile de PALLETE sau 1 536 (are 1 536 octeți). Virusul ZeroBug infectează fișierele EXE și .COM (exclusiv COMMAND). COM). Cît timp se află în memorie, ZeroBug va infecta orice fișier ce va fi executat și va determina afișarea dimensiunii exacte a fișierelor infectate, deși ele sînt mărite cu 1 536 octeți. După un anumit timp de la activarea virusului de pe ecran încep să dispară zerourile.

Viruşii CASCADE

Dimensiunea acestui virus poate fi 1 701 sau 1 704 octeți. În principiu toate varietățile infectează fișiere .COM mai mici de 63 800 octeți (inclusiv COMMAND .COM). Viruşii CASCADE se autoîncarcă în memorie utilizînd MCB-ul în mod aleator, se activează în lunile de toamnă (septembrie-decembrie) provocînd căderea, în rafale, a literelor de pe rîndurile superioare ale ecranului (numai la monitoare și adaptoare CGA, EGA, VGA). Acum ne explicăm celelalte denumiri ale virusului : Blackjack, Falling tears, Automn leaves, Herbist.

Virusul AIDS

Virusul AIDS, cunoscut și sub numele de HAHAHA sau TAUNT, infectează și suprascrie fișierele .EXE și .COM.

Cînd este activat virusul afișează mesajul :

Your computer now has AIDS"

și blochează calculatorul. Virusul AIDS suprascrie primii 13 Ko din fișierele original, distrugînd datele fără posibilitatea de reconstituire a acestora.

Viruşii YANKEE DOODLE

Există 26 de varietăți ale viruşilor Yankee Doodle, avînd mărimile între 1 300 și 3 300 octeți. Infectează fișierele .EXE și .COM (exclusiv COMMAND.COM). Se autoîncarcă folosind MCB-ul. După acționare va infecta orice fișier ce va fi executat. Dacă aveți norocul să fie activ virusul, la ora 17 veți putea asculta melodia Yankee Doodle.

Viruşii din această familie se recunosc între ei putîndu-se înlocui astfel versiunile vechi prin versiuni noi. De asemenea, sînt în măsură să depisteze programele de tip antivirus care încearcă să-i elimine.

Virusul 1701

După rularea unui program infectat cu virusul 1701, virusul va infecta fișierul COMMAND.COM sau următorul fișier .COM executat. Rămîne rezident după prima rulare pînă la reîncărcarea sistemului. Va infecta orice fișier .COM ce va fi rulat crescîndu-i dimensiunea cu exact 1 701 octeți.

Înlăturarea virusului se face prin înlocuirea tuturor fișierelor .COM infectate cu fișiere „curate" din dischete originale sau de salvare.

Este inutil să rebotezăm cu diverse nume secrete fișierul COMMAND.COM deoarece virusul 1 701 îl va recunoaște oricum.

8.2. VIRUȘI CARE INFECTEAZĂ SECTORUL DE AUTOÎNCĂRCARE/TABELA PARTIȚIILOR

Viruşii din această clasă au diferite reacții : se încarcă în memorie înaintea sistemului de operare, transferă sectorul de autoîncărcare în alt sector, mixează date,

micşorează volumul memoriei raportate de BIOS, distrug date etc.

Virusul PAKISTANI BRAIN

După Ping-Pong, virusul Pakistani Brain este al doilea virus care infectează sectorul de autoîncărcare. Este ceva mai mare decît Ping-Pong şi foloseşte trucuri inteligente.

Versiunea originală a virusului Brain nu face nici un rău, cu excepţia ocupării cîtorva sectoare. Dacă nu găseşte nici un sector neutilizat, nu infectează. Dar dacă găseşte numai un sector neutilizat îl va ocupa şi va „invada" şi sectoarele adiacente, sectoare ocupate cu date ale utilizatorului.

Asemănător celorlalţi viruşi din această clasă, virusul Brain infectează discul schimbînd sectorul de autoîncărcare asigurîndu-şi prin aceasta încărcarea sa în memorie înaintea sistemului de operare atunci cînd discul sistem este infectat. Sectoarele folosite de virus sînt marcate ca defecte.

Virusul Brain poate fi recunoscut cu uşurinţă după două semne distincte : *1)* sectorul de autoîncărcare al discului infectat conţine un mesaj dedicat tuturor viruşilor care nu sînt mai lungi decît el ; *2)* eticheta volumului în cazul unui disc infectat este „(C)Brain".

Pentru a-şi asigura protecţia, virusul are grijă ca orice acces de citire/scriere în sectorul de autoîncărcare să fie făcut la sectorul original. În acest fel utilizatorul nu poate scrie în primul sector pentru a distruge virusul.

Dacă virusul este rezident, un simplu A : sau DIR duc la infectarea altui disc.

În fig. 8.1 se prezintă sectorul de autoîncărcare infectat cu virusul Brain.

Virusul PING PONG

Primul virus care infectează sectorul de autoîncărcare, Ping-Pong, este numit şi Bouncing Ball sau Italian. Virusul ocupă primul sector din disc sau dischetă. Sectorul de autoîncărcare original este transferat într-un alt sector care va fi marcat ca defect de către virus.


```
Side 0, Cylinder 0, Sector 1

FAE94A01 34120005 09000100 00000000 57656C63 6F6D6520 œØJ4,      ......Welcome "
746F2074 68652020 44756E67 656F6E20 20202020 20202020 to the  Dungeon          "
28632920 31393836 20427261 696E1726 20416D6A 61647320 (c) 1986 Brain   Amjads "
28707674 29204C74 64202020 56495255 535F5348 4F452020 (pvt) Ltd   VIRUS_SHOE  "
5245434F 52442020 2076392E 30202020 44656469 63617465 RECORD   v9.0   Dedicate"
6420746F 20746865 2064796E 616D6963 206D656D 6F726965 d to the dynamic memorie"
73206F66 206D696C 6C696F6E 73206F66 20766972 75732077 s of millions of virus w"
686F2061 7265206E 6F206C6F 6E676572 20776974 68207573 ho are no longer with us"
20746F64 6179202D 20546861 6E6B7320 474F4F44 4E455353  today - Thanks GOODNESS"
21212020 20202020 20424557 41524520 4F462054 48452065 !!          BEWARE OF THE e"
722E2E56 49525553 20203A20 5C746869 73207072 6F677261 r..VIRUS  : \this progra"
6D206973 20636174 6368696E 67202020 20202070 726F6772 m is catching       progr"
616D2066 6F6C6C6F 77732061 66746572 20746865 7365206D am follows after these m"
65737365 6765732E 2E2E2E2E 20242340 25244021 21208CC8 esseges..... ¤#$%@!! _ "
8ED88ED0 BC00F0FB A0067CA2 097C8B0E 077C890E 0A7CE857 .Ø¸-ł.K¤|¢|.|"      |ŁW"
00B90500 BB007EE8 2A00E84B 0081C300 02E2F4A1 13042D07 .'.».~Ł*.ŁK. ^Ď¡-      "
00A31304 B106D3E0 8EC0BE00 7CBF0000 B90410FC F3A406B8 .£±*Ω.¤ł.|¿..'þ§§+      "
000250CB 5153B904 00518A36 097CB200 8B0E0A7C B80102CD .PQS'.Q'6|².. |+¯      "
137309B4 00CD1359 E2E7CD18 595B59C3 A00A7CFE C0A20A7C s ×.ŸÐLŸ[YÊ |ÊÊ |
3C0A751A C6060A7C 01A0097C FEC0A209 7C3C0275 09C60609 < u
```

Fig. 8.1. Un sector de autoîncărcare infectat cu virusul Brain

Acest lucru se poate verifica uşor cu un utilitar (de exemplu PCTOOLS) întrucît sectorul marcat ca defect conţine date.

Ori de cîte ori se foloseşte pentru încărcarea sistemului o dischetă infectată, virusul se încarcă în memorie, la sfîrşitul acestuia, în funcţie de mărimea raportată de BIOS şi micşorează valoarea memoriei cu 2Ko. Cînd virusul „se hotărăşte" să-şi anunţe prezenţa, un mic balon se va plimba vesel pe ecran.

Transmiterea virusului se face foarte uşor, inclusiv prin dischete care nu sînt dischete sistem. Primul acces făcut la o dischetă neinfectată (chiar o simplă comandă DIR) asigură infectarea acesteia dacă virusul este activ în sistem.

Sectorul de autoîncărcare de pe o dischetă infectat cu virusul PING PONG arată ca în fig. 8.2.

Virusul MISSPELLER

Sectorul de autoîncărcare infectat cu MisSpeller este în proporţie de 90% exact acelaşi ca la Ping-Pong fapt ce poate crea confuzii cu implicaţii neplăcute. De fapt, MissSpeller este o variantă israeliană a virusului Ping-Pong. Altfel îşi face MisSpeller simţită prezenţa în sis-

```
Side 0, Cylinder 0, Sector 1

EB1C904D 53444F53 332E3300 02020100 027000D0 02FD0200 ®■KSDOS3.3..p.-t.      "
09000200 000033C0 8ED0BC00 7C8ED8A1 13042D02 00A31304 ....3■.-½.|.■;-.£      "
B106D3E0 2DC0078E C0BE007C 8BFEB900 01F3A58E C80E1F88 ±®Ω-■.■½.|.'.δ¥.È       "
000032E4 CD138026 F87D808B 1EF97D0E 582D2000 8EC0E83C ..2Ñ■&‡}■.ø}X- .■Ł.<    "
008B1EF9 7D43B8C0 FF8EC0E8 2F0033C0 A2F77D8E D8A14C00 ..ø}C+■. .■Ł/.3■¢1}.■;L."
8B1E4E00 C7064C00 D07C8C0E 4E000E1F A32A7D89 1E2C7D8A .N.Ł.-|_N.£*}¨.}'       "
16F87DEA 007C0000 B80103EB 03B80102 9303061C 7C33D2F7 ‡}■.|..+?+■|3®1-        "
36187CFE C28AEA33 D2F7361A 7CB106D2 E40AE58B C886E98A 6|'■3®16|±®H ■.¯ø'      "
F28BC38A 16F87DEB 0080CD13 730158C3 1E065053 51520E1F d.^‡}».■§XÊSQR          "
0E07F606 F77D0175 4280FC02 753D3816 F87D8816 F87D7522 ÿÿ}uD■pu=8‡}"‡}u"       "
32E4CD1A F6C67F75 0AF6C2F0 750552E8 B1015A8B CA2B16B0 2H"ÿÿ ÿÿuRL±Z.‡°        "
7E890EB0 7E83EA24 7211B00E F77D0156 57E81200 5F5E8026 ~"°~^Ω«■1}VWL.,^■&     "
F77DFE5A 595B5807 1FEA8303 00C8B801 02B600B9 0100E88A 1}ZY{X■^.÷■.'.È¯        "
FFF606F8 7D807423 BEBE81B9 0400807C 0401740C 807C0404 .ÿ‡■t■‡‡¹'.■|t ■‡      "
740683C6 10E2EFC3 8B148B4C 02B80102 E860FFBE 0280BF02 t¯Ďh^.L+Ł'.‡■;          "
7C891C00 F3A4813E FC815713 7515803E FB810073 0DA1F581 |'.δ$'>þ'Wu■>'.s ;'     "
A3F57D8B 36F981E9 0801C381 3E0B8000 0275F780 3B0D8002 £}.6ø'Ø'> ■.u1■> ■      "
72F08B0E 0E80A010 8098F726 168003C8 B82000F7 26118005 rK.■■■1■■÷   .1■■       "
FF01BBC0 02F7F303 C8890EF5 7DA1137C 2B06F57D 8A1E0D7C .».1■'};|+}'     |       "
33D232FF F7F3408B F88026F7 7DFB3DF0 0F760580 0EF77D04 3®2.1ø@.‡■‡1}=kv■1}     "
BE01003B 1E0E7C4B 891EF37D C606B27E FEEB0D01 000C0001 ¾.¸|K"ó}ʸ~º .          "
01680100 571355AA          Press Enter to continue    h.WU"                  "
```

Fig. 8.2. Sectorul de autoîncărcare de pe o dischetă infectat cu virusul Ping Pong

tem : în timp ce se autoîncarcă face mixarea datelor ce se transferă spre portul paralel.

Asemănător cu virusul Ping Pong va infecta sectorul de autoîncărcare din disc și se încarcă în memorie, la sfîrșitul acesteia, micșorînd valoarea memoriei cu 2K.

Virusul STONED

Acest virus a fost descoperit în septembrie 1989 și este cunoscut și sub numele de MARIJUANA. Infectează sectorul de autoîncărcare al dischetelor și tabela de alocare a partițiilor de la discuri la hard. Spre deosebire de alți viruși care infectează sectorul de autoîncărcare, virusul Stoned nu se trădează prin marcarea ca defect a sectorului in care mută sectorul de autoîncărcare original. Dacă încărcarea sistemului este făcută de pe o dischetă infectată, adeseori va fi emis mesajul :

„Your PC is now Stoned — LEGALISE MARIJUANA !"

Virusul se autoîncarcă la sfîrșitul memoriei și diminuează cu 2Ko volumul memoriei raportate de BIOS (la


```
Side 0, Cylinder 0, Sector 1

EA0500C0 07E99900 02830300 C8E40080 9F007C00 001E5080 Ω.■■}.^.Ä.■■.|..P■.....'
FC027217 80FC0473 120AD275 0E33C08E D8A03F04 A8017503 pr■■ps,®u3■.■■?¨u........"
E8070058 1F2EFF2E 09005351 52065657 BE0400B8 01020E07 Ł.X...     .SQRVW¼.+     "
BB000233 C98BD141 9C2EFF1E 0900730E 33C09C2E FF1E0900 ».3.'A■...s3■■.. .        "
4E75E0EB 359033F6 BF0002FC 0E1FAD3B 057506AD 3B450274 NuΩ95■3ÿ¿.þ‡;u‡;Et        "
21B80103 BE0002B1 03B6019C 2EFF1E09 00720FB8 010333DB !÷».±■■.. .r+3■          "
B10133D2 9C2EFF1E 09005F5E 075A595B C333C08E D8FA8ED0 ±3®■..     ._^ZY{3■.■■.-"
BC007CFB A14C00A3 097CA14E 00A30B7C A1130448 48A31304 ¼.|;L.£   |;N.£ |;HH£   "
B106D3E0 8EC0A30F 7CB81500 A34C008C 064E00B9 B8010E1F ±®Ω.■£|+.£L._N.'+        "
33F68BFE FCF3A42E FF2E0D00 B80000CD 1333C08E C0B80102 3ÿ.ÞÞ$... .+..Ø■.■+     "
BB007C2E 803E0800 00740BB9 0700BA80 00CD13EB 4990B903 ».|.■>..t '.'■.ºI■'     "
00BA0001 CD13723E 26F6066C 04077512 BE89010E 1FAC0AC0 .".Ø>&ÿlu, + ■           "
7408B40E B700CD10 EBF30E07 B80102BB 0002B101 BA8000CD t×.·.Ø‡×.±"■.¯           "
1372130E 1FBE0002 BF0000AD 3B057511 AD3B4502 750B2EC6 r‡.¿..«;u‡;Eu .¯          "
06080000 2EFF2E11 002EC606 080002B8 0103BB00 02B90700 .......¯+».'.            "
BA8000CD 1372DF0E 1F0E07BE BE03BFBE 01B94202 F3A4B801 "■.ºÞ‡‡¿‡'ÞÞ$+          "
0333DBFE C1CD13EB C507596F 75722050 43206973 206E6F77 3■Ø¯   Your PC is now"
2053746F 6E656421 070D0A0A 004C4547 414C4953 45204D41  Stoned!     .LEGALISE MA"
52494A55 414E4121 00000000 00008001 01000103 11971100 RIJUANA!       .....■.■."
00004F28 00000000 01980503 51306028 0000A428 00000000 ..O(....Q0`(..$(........"
00000000 00000000 00000000 00000000 00000000 00000000 ........................"
00000000 000055AA          Press Enter to continue    ......U"                "
```

Fig. 8.3. Tabela partițiilor infectată cu virusul Stoned

fel cu Ping Pong și MisSpeller). Sectorul de autoîncărcare original este salvat la sectorul 3 al pistei 0, pe fața 1 în cazul dischetelor și la sectorul 7, pista 0, fața 0, în cazul harddiscurilor. Trebuie reținut că nu salvează datele acestor sectoare înainte de infecție, producînd „încurcături" pentru director și tabela de alocare a fișierelor.

În cazul harddiscului virusul Stoned infectează tabela partițiilor (fig. 8.3).

Virusul YALE

Virusul Yale infectează numai dischetele de 360 K. Este cunoscut și sub numele de Alameda și Merritt. Atunci cînd în unitate este o dischetă „sănătoasă" și se face încărcarea la cald (ALT+CTRL+DEL), discheta va fi infectată. Sectorul de autoîncărcare original este depus la sectorul 8 al pistei 39, fața 0, distrugînd datele care eventual erau înainte de infectare în acest sector. Alte neplăceri sau necazuri nu produce.

9.

PROTECŢIA CONTRA ABUZURILOR... INFORMATICE

9.1. CE ESTE UN SISTEM DE PROTECŢIE

Contra răufăcătorilor umani sau informatici trebuie găsite căile şi mijloacele de anihilare sau cel puţin de descurajare a acţiunilor lor. De asemenea, protecţia informaţiei, sub diferitele ei aspecte, se poate realiza utilizînd mijloace, metode, modele şi... chiar viruşi.

Este de altfel binecunoscută din biologie situaţia în care viruşii atacă chiar bacterii distrugîndu-le !

În orice sistem informatic datele şi programele sînt componente vitale, care trebuie protejate de pierdere, distrugere, intruziune voită sau accidentală.

Prin construcţia unui calculator compatibil IBM PC şi a sistemului de operare, oricine poate citi, scrie, copia sau şterge orice informaţie din memoria sa internă sau externă. Toate calculatoarele trebuie protejate prin programe speciale de protecţie.

Sistemele de protecţie trebuie să asigure programele şi datele împotriva următoarelor acţiuni :

— pierderi accidentale ;
— acces neautorizat la date ;
— copie neautorizată a programelor ;
— fraudă pe calculator (furt de date sau alterarea lor, furturi de servicii etc.) ;
— maladii software, viruşi ai calculatoarelor ;
— utilizatori începători.

Pierderile accidentale rezultă în general din căderile de tensiune, defectarea unităţilor de discuri etc. Una din soluţiile cele mai comode şi mai eficiente pentru astfel

de probleme este utilizarea unor unităţi de medii magnetice şi a unui utilitar eficient de salvare.

Accesul neautorizat la date se referă în primul rînd la informaţiile financiare, de personal, date secrete (militare, particulare) etc. Acesta se poate evita prin introducerea unor mecanisme de parolă care să limiteze accesul diferiţilor utilizatori la diferite niveluri ale informaţiilor, oprind inclusiv copierea neautorizată a fişierelor.

Frauda pe calculator este o acţiune care s-a extins în multe ţări, în ciuda unui sistem legislativ foarte bine pus la punct. Astfel, în S.U.A., după o statistică de la Centrul naţional de fraudă pe calculator, frauda de acest tip se împarte astfel :

— furt de bani : 45% ;
— furt de date şi/sau programe : 16% ;
— distrugerea programelor : 15% ;
— alterări de date : 12% ;
— furt de servicii : 10% ;
— restul : 2%.

Valoarea medie a unei fraude, în dolari, este : 93 600 pentru distrugeri de date şi/sau sisteme ; 55 166 pentru furturi de programe şi de date ; 10 517 pentru furt de bani. Anual, furtul de bani prin calculator se situează între 3 şi 5 miliarde.

Un sistem complet de protecţie trebuie să conţină cinci componente importante :

1) identificarea : identificarea şi autoidentificarea utilizatorilor prin parole, cărţi de identificare, dispozitive de decodificare, de citire a amprentei ; acestea pot fi dispozitive fizice (foarte scumpe) sau programe care să realizeze aceste funcţii ;

2) izolarea : separarea datelor şi/sau programelor între utilizatori ; una din metodele cele mai eficiente este codificarea fişierelor cu o parolă specifică utilizatorului ;

3) controlul accesului : specificarea şi implementarea regulilor de acces şi a relaţiilor între utilizatori şi resurse ;

4) evidenţa accesului : urmărirea folosirii resurselor sistemului de către utilizatori ; evidenţa este necesară

din diverse motive : financiare, de securitate a datelor, de refacere a unor date în cazul unor distrugeri de informații etc. ;

5) *software* : menținerea unor copii separate ale datelor și programelor ; aceste copii pot avea un format special care fac inaccesibile informațiile pentru alte persoane.

În general metodele de protecție se clasifică în două mari grupe :

1) metoda care împiedică piraţii informatici ;

2) metode care detectează atacatorii.

Fiecare din aceste grupe se poate realiza prin software, hardware sau printr-o combinație hard și soft.

Cea mai simplă cale de protecție este ... automutilarea care are la bază regula următoare : *Remediul contra migrenei constă în tăierea capului.*

Aplicată la informatică, regula arată că numai calculatorul închis se găsește protejat 100% împotriva virușilor. La polul opus, un sistem complet deschis nu oferă deloc protecție. De aceea trebuie căutată o cale intermediară între aceste două extreme.

Calculatoarele personale sînt în mod special vulnerabile pentru că nu au deloc protecție hardware a memoriei : un program ce se execută poate avea acces oriunde, în memorie sau pe disc. Ca să nu mai vorbim de rețele de PC-uri care sînt și mai vulnerabile.

Ce se poate face ? Vom enumera în continuare o serie de „instrumente", software, dar și de altă natură care caută să împiedice accesul oricui, oriunde și oricum în avuția echipamentelor.

9.2. MIJLOACE DE ASIGURARE A PROTECȚIEI

În categoria mijloacelor de asigurare a protecției și confidențialității informației se includ :

a) *mijloace fizice* — constituie primul aliniament de apărare a celor mai importante elemente ale unui sistem de calcul : baza de date, programe, memorii, dispozitive de intrare-ieșire etc.

b) *mijloace tehnice* — care sînt reprezentate prin aparate electronice, electronooptice, electromecanice etc., capabile să îndeplinească funcții de protecție. Din categoria acestora, cea mai mare răspîndire o au : 1) registrele speciale pentru păstrarea elementelor de protecție (parole, coduri, nivelul de secretizare etc.) ; 2) generatoarele de coduri ; 3) dispozitive pentru citirea codurilor de pe purtătorii de informație care au ca efect punerea în funcțiune a unor mijloace tehnice, blocarea/deblocarea unor părți de echipamente ; 4) dispozitive de recunoaștere a vocii, amprentelor etc. ; 5) scheme pentru întreruperea transmiterii informației în linie ; 6) dispozitive de cifrare și descifrare.

c) *mijloace de programare* — incluse în sistemul de operare sau programe separate. Aceste programe speciale asigură funcția de protecție prin : 1) identificarea mijloacelor tehnice, problemelor, utilizatorilor sau a fișierelor folosite ; 2) ștergerea informației din memorie după utilizarea ei ; 3) semnalizarea audio/video în cazurile neautorizate ; 4) cifrarea informației etc.

d) *mijloace criptologice* — care constituie cea mai sigură modalitate de protecție și securitate a informației. Cifrarea se poate face fie pe calea programelor (care este cea mai ieftină), fie cu ajutorul unei aparaturi speciale (recomandabilă cînd volumul este foarte mare).

e) *mijloace (măsuri) organizatorice* — care cuprind : 1) proiectarea, construcția și echiparea centrelor de calcul sau a posturilor de lucru pentru a exclude influența catastrofelor naturale (cutremure, incendii, inundații etc.) ; 2) alegerea atentă și pregătirea profesională corespunzătoare a personalului de serviciu ; 3) organizarea unui regim sigur de control a accesului ; 4) organizarea păstrării și folosirii documentelor și purtătorilor de informații ; 5) controlul sever al introducerii modificărilor în sistemul de operare etc.

f) *măsuri juridice* — care cuprind acte normative care controlează și reglementează procesul prelucrării și folosirii informației.

9.3. MODELE DE ASIGURARE A PROTECȚIEI

Unul dintre primele modele de descriere a mecanismului de protecție este cel al lui Hoffman, care consideră patru tipuri de obiecte protejate : utilizatori *(u)*, programe *(p)*, terminale *(t)* și fișiere *(f)*. Fiecare obiect este descris cu un set de patru caracteristici de securitate *(A, C, D, R)* : *1)* autoritatea *A*, care este un scalar format din mulțimea ierarhizată pe caracteristici ale securității : nesecret, confidențial, secret, strict secret ; *2)* categoria *C*, care este un set de compartimente specifice în care se grupează datele, independent de nivelul de autoritate, cum ar fi : acces limitat, acces cu aprobare ; *3)* dreptul *D* reprezintă grupa de utilizatori care au acces la un obiect dat ; *4)* regimul *R* este o mulțime de moduri de acces la un obiect dat : citire, actualizare, execuție program.

Fie $U = \{u\}$ mulțimea utilizatorilor cunoscuți în sistem și $D(i)$ — mulțimea utilizatorilor cu drept de acces la obiectul *i*.

Regulile după care lucrează modelul sînt :

1) utilizatorul *u* are acces la terminalul *t* dacă și numai dacă $u \in U$;

2) utilizatorul *u* are acces la terminalul *t* dacă și numai dacă $u \in D(t)$;

3) utilizatorul *u* are acces la fișierul *f* dacă și numai dacă $A(p) \geqslant A(f)$, $C(p) \geqslant C(f)$, $u \in D(f)$, adică :

— privilegiul programului *p* este mai mare sau egal cu privilegiul fișierului ;

— utilizatorul face parte din mulțimea $D(f)$.

Un alt model de autorizare al accesului îl constituie modelul Kent, care are la bază cinci dimensiuni : *A*-împuterniciri, *U*-utilizatori, *E*-operații, *R*-resurse, *S*-stări. În acest caz, spațiul de securitate este dat de produsul cartezian $A \times U \times E \times R \times S$.

Accesul *q* la date este considerat ca o serie de cereri ale utilizatorilor *U* pentru operațiile *e* la resursele *R* într-un moment cînd sistemul se află în starea *s* :

$$q = (u, e, \mathfrak{R}, s), \quad u \in U, \quad e \in E, \quad s \in S, \quad \mathfrak{R}. \in R$$

În acest fel cererea de acces reprezintă un subspațiu al unei proiecții cu patru dimensiuni a spațiului de securitate.

Pentru o cerere *q*, o mulțime *U* de grupuri de utilizatori, o mulțime *R* de resurse elementare și un set *P* de specificații de autorizare, procesul de organizare a accesului constă din pașii :

P1 : se apelează programele auxiliare necesare luării deciziei preliminare ;

P2 : se determină din *U* acele grupe de utilizatori din care face parte și *u* ;

P3 : se aleg din *P* acele specificații de autorizare cărora le corespund grupurile de utilizatori stabilite ; mulțimea de autorizări *F(u)* se numește *privilegiul utilizatorului u* ;

P4 : se determină din *P* setul *F(e)* cu drepturile care stabilesc *e* ca operație de bază elementară ; setul acesta se numește *privilegiul operației e* ;

P5 : se determină din *P* setul $F(\mathfrak{R})$ — *privilegiul resursei elementare* $\mathfrak{R}$, format din împuternicirile care definesc submulțimi de resurse din $\tilde{\mathfrak{R}}$, care au elemente comune cu resursele cerute ; autorizările comune celor trei privilegii constituie domeniul de autorizare *D(s)* al cererii *q* :

$$D(q) = F(u) \cap F(e) \cap F(\mathfrak{R})$$

P6 : se asigură că resursa cerută $\mathfrak{R}$ este complet acoperită de *D(q)*, adică fiecare element al lui *R* trebuie să fie conținut într-o resursă elementară care este specificată în domeniul *D(q)* ;

P7 : mulțimea *D(q)* este proporțională în clase de echivalență bazate pe relația : *două autorizări sînt de aceeași clasă dacă și numai dacă aceasta specifică aceeași resursă elementară* ; pentru fiecare clasă se calculează o operație logică SAU/ȘI, cu condițiie de acces ale elementelor sale. Efectul se asigură prin efectuarea operației logice SAU/ȘI asupra tuturor grupelor de utilizatori care conțin utilizatorul *u* în raport cu cererea *q* ;

P8 : se calculează condiția de acces efectiv asupra tuturor resurselor elementare care se suprapun peste resursa unitară cerută ;

P9 : se evaluează condiția de acces efectiv și se ia decizia de acces (se permite accesul la *R* dacă *R* este acoperită sau se refuză accesul în caz contrar).

P10 : dacă decizia de acces este de aprobare se execută programele ce decurg din condiția de acces ; în caz contrar se execută programele ce decurg din condiția de refuz.

Exemple de metode destinate asigurării protecției sînt : criptografia, parola, ținerea și controlul jurnalelor de înregistrare, ștergerea resturilor de date din memoria operativă etc.

Tot Hoffman consideră că se poate măsura gradul protejării sistemului folosind variabilele lingvistice care iau valori sub forma cuvintelor și nu a cifrelor.

9.4. PROTECȚIE, CONFIDENȚIALITATE, SIGURANȚĂ

Nevoia de protecție diferă foarte mult și este în funcție de natura aplicației și domeniul de utilizare. Măsurile de asigurare a protecției, confidentialității, siguranței etc. necesită măsuri multiple și complexe împotriva atentatorilor. Aceștia pot atenta la informația ca atare, la sistemele de operare, fișiere, baze de date și sisteme teleinformatice. De aceea, o trecere în revistă a măsurilor corespunzătoare este necesară.

Protecția informației. Evoluția tehnicilor de penetrație neautorizată, utilizînd programe dintre cele mai rafinate, impune ca necesară o abordare metodică a proiectării și implementării mecanismelor de protecție. Aici însă vom face numai o prezentare sumară a acestora. Astfel, pentru protecția informației avem la dispoziție :

— mecanismul de protecție prin decriptori, bazat pe registrul decriptor cu cele două componente ale sale : baza (adresa cea mai mică pe care se poate utiliza un program) și limita (numărul de locații la care are acces programul) ; mecanismul se poate implementa prin listă sau prin tichet ;

— mecanismul de protecție prin capacități (drept de acces) care are două componente : *a)* de autentificare, care controlează accesul unui utilizator la capacitatea corespunzătoare catalogului său ; *b)* bazată pe capacități care controlează accesul la diferite segmente protejate ale sistemului ; controlul propagării drepturilor de acces trebuie să evite posibilitatea ca un utilizator, după ce a primit o capacitate pentru acces la un anumit fișier care nu îi aparține, să facă copii ale acestuia și să le distribuie către alți utilizatori fără permisiunea proprietarului obiectului protejat ;

— mecanism pe protecție prin lista de control a accesului pentru un grup de utilizatori ; alocarea spațiului pentru listă variază în timp și creează unele probleme la implementare.

Protecția sistemelor de operare. Calculatoarele și programele de firmă (sistemele de operare, sistemele pentru gestiunea bazelor de date etc.) conțin unele mijloace speciale destinate protecției, și anume :

— scheme de întrerupere ;

— blocuri pentru protecția memoriei ;

— registre de protecție etc.

În toate sistemele de operare accesul la informație este controlat cu ajutorul unor parole speciale fiind prevăzută și separarea precisă a resurselor sistemului între programe și procesele aflate simultan în execuție în sistemul de calcul.

Harrison și Ruzzo au elaborat modelul formal al unui sistem de protecție bazat pe matrice de acces și au abordat problema siguranței, adică determinarea, într-o situație dată, a posibilității unui subiect de a putea obține un anumit drept de acces la un obiect.

Un alt model de protecție este cel bazat pe grafuri care folosește regulile de „primire-acordare". Obiectele din cadrul modelului sînt : active (utilizatori sau programe), pasive (fișierele) și obiecte despre care nu se poate ști că sînt pasive sau active. Graful finit, numit graf de protecție, este construit cu aceste obiecte conectate prin arce etichetate.

În sistemul de operare UNIX s-a implementat un sistem simplu de protecție în care fiecare fișier are un singur. proprietar. Acesta specifică propriile lui drepturi (citire, scriere, execută), precum și ale celorlalți utilizatori care fac parte din grup. Sistemul nu face distincție între subiecți exceptînd calitatea de proprietar-neproprietar.

Protecția și securitatea fișierelor de date. Cea mai simplă metodă de asigurare a integrității datelor și refacerii fișierelor în cazul cînd acestea au fost alterate o reprezintă *tehnica copiilor de siguranță* care presupune salvarea la anumite momente prestabilite a fișierelor de asigurare. Salvarea poate fi efectuată înainte sau după efectuarea modificărilor fișierelor de asigurat.

În situația sistemelor complexe prevăzute în regim de teleprelucrare sau în situația unor aplicații ce durează mai mult timp se creează fișierele numite *cozi de mesaje*, care vor conține informații referitoare la starea sistemului în anumite momente de prelucrare. Pentru aceasta analistul prevede puncte de repriză (reluare), adică momentele de timp la care se reține starea sistemului. Pe lîngă fișierul copie de siguranță se creează și fișierele : *a)* de urmărire generală a actualizărilor din momentul în care începe derularea aplicației și pînă se încheie aceasta ; *b)* de urmărire a actualizărilor între două puncte de repriză.

În cazul *fișierelor cu legături* trebuie avută în vedere asigurarea unei copii a fișierului cu legături sau a unei proceduri capabile să refacă acest fișier.

Pentru *fișierele inverse* integritatea este asigurată cu ușurință dacă grupul de fișiere este în întregime simetric.

Protecția fișierelor se poate realiza cu mai mult succes dacă se utilizează *metode criptografice*. Pentru aceasta se utilizează două clase importante de transformări criptografice :

— cifrare controlată de sistem : datele, indiferent de utilizator, sînt tratate la fel ; cheile de cifrare sînt memorate în sistem ;

— cifrarea controlată de utilizator : utilizatorul va furniza cheia cînd dorește acces la fișier.

Modalitățile de implementare a acestei protecții sînt : *a)* protecția compartimentală și independentă de date ; *b)* protecția ierarhică și independentă de date. Pentru fiecare din acestea există algoritmi precum : algoritmul profil-utilizator (cheile sînt memorate în sistem), algoritmul „articol cu chei" (la începutul fiecărui fișier se adaugă un articol cu chei și un articol de validare) etc.

Protecția și securitatea bazelor de date. Asigurarea integrității datelor are loc prin utilizarea *copiilor de siguranță* și a *jurnalelor de urmărire a actualizărilor*. Crearea jurnalelor se realizează la momente prestabilite de utilizator iar copiile de siguranță sînt realizate de administratorul bazei de date tot la momente prestabilite.

Asigurarea secretului datelor din bazele de date este realizată prin *tehnica cheilor de acces* și pentru această tehnică există mai multe variante. Una dintre ele este aceea în care administratorul bazei oferă fiecărui utilizator o cheie de acces. O altă variantă constă în a schimba cheia la fiecare sesiune de lucru, cheie aflată într-o listă de chei de acces. Tot ca variantă se poate reține faptul că se folosesc anumiți algoritmi sau un anumit tip de dialog la care utilizatorul trebuie să dea numai răspunsuri corecte la întrebările puse de sistem.

Controlul accesului la o bază de date se poate realiza pa baza mai multor modele : model bazat pe matricea de acces, extensia modelului bazat pe matricea de acces, modele multinivel, securitatea compartimentală etc.

Protecția și securitatea rețelelor de teleprelucrare. Chiar dacă este posibilă asigurarea unui înalt nivel de securitate fizică pentru calculatoarele din rețea, de cele mai multe ori este imposibil să se asigure o protecție fizică corespunzătoare pentru linia de telecomunicații. Ca urmare, utilizatorii neautorizați se pot conecta pe liniile de date sau se pot interpune cu ajutorul unui calculator între două noduri ale rețelei.

Aceste posibilități de atac asupra unei rețele provoacă serioase daune prin afectarea caracterului privat al transmisiilor, introducerea unor informații false sau trunchiate, falsificarea identității unor calculatoare, terminale sau utilizatori.

Pentru a „ataca" o rețea de calculatoare trebuie să cunoaștem cel puțin patru noțiuni importante :

a) *Identificarea* reprezintă recunoașterea unui număr, nume sau cod unic care este afectat unei persoane, proces sau echipament. Identificarea este necesară în procesul de control al accesului la un echipament.

b) *Autentificarea*. Aceasta vizează două aspecte : autentificarea utilizatorului de către sistem sau a sistemului de către utilizator.

În acest scop se pot utiliza mai multe metode :

— prin parolă (simplă sau intercalată cu caractere speciale), care cere utilizatorului introducerea unui șir de caractere ce va fi verificat de calculator și, numai dacă șirul este corect, se permite accesul ;

— prin chestionar, cînd utilizatorul trebuie să dea răspunsuri corecte la toate întrebările puse de calculator și, numai în acest caz, i se permite accesul ;

— prin proceduri, cînd sistemul cere utilizatorului să execute corect un algoritm și numai după aceea i se acordă accesul ;

— prin metode criptografice, folosind un protocol de autentificare reciprocă utilizînd o cheie prearanjată ;

— prin metode fizice, cînd se verifică anumite caracteristici fizice ale utilizatorului, terminalului sau calculatorului.

c) *Autorizarea*. Este acțiunea prin care se verifică dacă o persoană/proces au un anumit tip de drept de acces la o anumită resursă protejată. De obicei, pentru autorizare se consultă matricea de autorizare care este „inima" sistemului de securitate.

d) *Controlul accesului*. Acesta se realizează prin verificarea datelor furnizate de utilizator cu anumite informații de autentificare și cu informații privind tipurile de acces permise asupra resurselor.

9.5. PROTECȚIA CU AJUTORUL... VIRUȘILOR

Dezvoltările precedente par să sugereze că putem utiliza un virus „gentil" pentru a distruge alt virus. Posibilitățile concrete în acest domeniu sînt multiple.

Virusul „gentil" este introdus în sistem. Virusul „rău" recunoaște victimele (programe infectate) programului gentil și nu le mai atacă. Pentru a pune în aplicare acest gen de proiect, trebuie cunoscut în detaliu un virus, căci o dată ce amprenta a fost decodată este relativ ușor să detectăm programele infectate.

Iată în continuare, un alt tip de protecție prin virus. Se introduce un virus în sistem. Obligațiile virusului constau în detectarea software-ului atacat deja. Pentru fiecare program bolnav, virusul calculează una sau mai multe chei de control care se înregistrează. Cînd lansăm programul, virusul devine activ și se calculează cheile. Dacă există vreo modificare, de exemplu a infecției printr-un alt virus, atunci cheile calculate diferă de cheile de referință și se va afișa un mesaj de avertizare sau pur și simplu programul este întrerupt semnalizîndu-se aceasta sonor sau vizual.

În ciuda competenței lor aparente, aceste metode sînt departe de a garanta o protecție rezonabilă. Publicațiile de specialitate tipăresc studii ce descriu mecanisme de acest tip. Așa cum este imposibil de a aștepta programe cu cod mai puțin virulent, la fel de imposibilă este protecția eficace contra virușilor. Argumentele sînt evidente :

— toate modificările de software anulează garanția fabricantului (producătorului) :

— toate funcțiile protectoare realizate cu ajutorul virușilor pot fi egalizate de tehnicile curente de programare ;

— o protecție ce se găsește în interiorul software-ului amenință foarte bine detectarea și decodarea ;

— dezvoltările de viruși protectori pot conduce la prejudicii sensibile !

Un alt mod de luptă este „inocularea" (vaccinarea) unui program-obiect prin plasarea unei rutine de autentificare în punctul de intrare. Aceasta poate fi învinsă de un virus mai dibaci ce invadează acest punct.

Un alt program anticorp imaginat și creat de informaticieni oferă și el protecție, dar limitată. Un astfel de program examinează fișierele obiect și semnalează utili-

zatorului existența unui virus cunoscut. Unii îndepărtează virusul din programul infectat, dar ce se întîmplă cu cei noi ?

O altă schemă de protecție mai bună se bazează pe semnătura digitală a fișierului-obiect. În ce constă aceasta ? Cînd un program este introdus în memorie se creează un autentificator prin calcularea unei sume de control ce depinde de toți biții fișierului, care este apoi codificată cu o cheie secretă. Autentificatorul este deblocat numai prin aplicarea acestei chei. Un program infectat nu mai poate fi utilizat deoarece nu mai răspunde corect la această cheie și atunci printr-un anumit semnal, mesaj sau acțiune utilizatorul este avertizat.

Capcanele mascate, un alt mod de a învinge maladia, sînt de obicei conținute în programele cumpărate sau închiriate și sînt folosite la pedepsirea copierii ilegale a anumitor programe protejate. Cea mai cunoscută protecție de acest fel este cea a compilatorului dBase Clipper. Firma producătorului a livrat în Europa variante care după 750 compilări se autodistrug. Produsul-program FoxBase are în interiorul său o capcană mascată. Aceasta compară numărul de serie al programului cu informația de identificare înregistrată pe discul Winchester la instalare. Dacă aceasta nu corespunde, programul execută niște instrucțiuni interne care aduc FoxBase-ul la nivelul lui dBase II.

Aceste mecanisme pot limita numărul de copieri ale programelor, numărul de lansări în execuție, posibilitatea lansării fără o anumită dischetă care nu poate fi copiată cu funcții sistem obișnuite etc. Mecanismele utilizează programe speciale de copiere, parole, dischete care nu respectă formatarea standard, chei multiple de acces, dischete cu defecte hard predefinite etc. Toate acestea au scopul de a limita utilizarea produsului la cei care l-au cumpărat de la firmă.

Merită de asemenea amintită o metodă mai specială pentru apărarea intereselor firmelor producătoare și distribuitoare de discuri flexibile de mare capacitate. Pe ambele fețe ale discurilor de acest tip există cîte un sector ascuns care nu permite formatarea acestuia la 360 Kocteți.

Există la ora actuală o serie de sisteme care ne asigură protecția.

Apariția virusului WDEF, considerat ca unul dintre cei mai inteligenți, a condus la intensificarea eforturilor de elaborare a programelor de protecție contra acestuia. Firmele Symantec și Microcom, considerate ca standard în lupta antivirus, au elaborat deja două astfel de programe : SAM (Symantec Antivirus for Macintosh) și VIREX.

Virusul WDEF se multiplică la deschiderea unei ferestre sau la mutarea ei. De asemenea, el ocolește toate schemele de protecție antivirus cunoscute pînă acum și se „lipește" de resursele de definire fereastră din fișierul desktop.

Produsul VIREX se poate actualiza pentru orice nouă clasă de virus.

VIRUSAFE+ este cel mai avansat sistem de protecție contra virușilor informatici. De asemenea, este cel mai eficace remediu de luptă. Sistemul are următoarele caracteristici :

— detectează și identifică toți virușii care infectează boot-ul, partițiile discurilor, fișierele-program și memoria microcalculatorului ;

— verfică la cerere programele (mărime, structură, check-sums) ;

— imunizează, neutralizează și elimină virușii ;

— el funcționează pe microcalculatorul IBM PC/AT/XT, IMB PS/2 și compatibile cu acestea sub sistemul de operare MS-DOS/PC-DOS pe dischete de 5"1/4 sau 3"1/2.

Dintre virușii care pot fi detectați și anihilați de către VIRUSATE+ amintim : Ping-pong, Alabama, Brain...

AS (Alteration Searcher) este un program care atacă o singură caracteristică comună a virușilor : detectează modificările efectuate de viruși și restabilește informația inițială sub o formă nouă. El controlează :

— modificările programelor și datelor ;

— programele și datele noi introduse ;

— programele și datele șterse sau substituite.

Controlul constă în a testa dacă starea memorie de masă a fost schimbată. El recunoaște, independent de op-

ţiuni, ansamblul interiorului unui fişier. Modificările găsite le înregistrează în fişierul LOG sub formă editabilă şi imprimabilă.

Programul este scris complet în limbaj de asamblare.

Unii producători au început să încorporeze în produsele lor scheme de protecţie antivirus prin care se verifică continuu rularea corectă a programului. O astfel de schemă este implementată, deocamdată, doar de Mac Write 2. Însă acestă metodă nu este încă viabilă.

Observaţie ! Utilizarea virusilor contra virusilor este totuşi o metodă puţin sigură şi totodată primejdioasă !

10.

LUPTA CONTINUĂ...

Pînă în prezent au pătruns şi la noi în ţară mai multe tipuri de viruşi informatici. Urmările acestor pătrunderi sînt de la o simplă glumă (scrierea unor mesaje pe ecran), pînă la ştergerea sau distrugerea fişierelor existente pe disc, blocarea sistemelor etc.

Pentru a micşora posibilitatea infectării există anumite reguli elementare de „igienă" a calculatorului personale :

— nu se lansează în execuţie (fără să fie instalat un vaccin) programe care nu sînt obţinute din surse sigure ;

— trebuie păstrate copii pe disc flexibil sau pe bandă ale programelor originale şi ale datelor importante ;

— nu se încarcă sistemul de operare de pe discuri flexibile neverificate ;

— se recomandă utilizarea unor programe specializate de „vaccinare" care să oprească virusul în momentul intrării lui în sistem ;

— nu ştergeţi un fişier decît după ce v-aţi convins că... este al dumneavoastră.

Din moment ce putem atribui calculatoarelor o anumită sensibilitate, este firesc să vorbim şi de un sistem imunitar pe care omul trebuie să i-l implanteze. Astfel, un neologism care circulă frecvent în toate centrele de calcul din lume este PERIMATICA — o disciplină a exploatării în siguranţă a calculatoarelor. Pentru o fiabilitate corespunzătoare trebuie studiate şi aplicate „tratamente" ce vizează securitatea sistemelor : sisteme de pro-

tecție și siguranță în flux, protecție împotriva incendiilor și supraîncălzirilor, filtrarea impurităților aerului din incinta respectivă etc.

Însă, perimatica s-a confruntat și luptă continuu cu un nou adversar inocent dar destul de agresiv : virusul informatic. Există o diversitate de atentate la imunitatea calculatoarelor, împotriva cărora perimatica este chemată să confecționeze vaccinuri.

Nu au trecut decît cîteva decenii de la implementarea primelor mecanisme de tip viral în lumea informaticii.

Aventura lumii vii, care ne-a dominat și ne domină gîndirea de cîteva secole, nu avea cum să nu-și pună amprenta asupra creațiilor majore ale gîndirii umane. Și informatica este fără îndoială una dintre cele mai spectaculoase cuceriri ale minții omenești.

Deocamdată virusul informatic rămîne doar o formă imperfectă a semenului său biologic, care este rodul unei evoluții spre perfecțiune care durează de mii și mii de ani. Multe din secretele cu care natura a înzestrat organismele vii rămîn încă păzite cu strășnicie. Poate tentativele de experimentare în informatică a unor mecanisme specifice lumii vii să faciliteze o înțelegere a lor într-o deplină coerență.

O întrebare tulburătoare se ridica însă tot mai pregnant. Este oare posibil ca acest element nou, pătruns în lumea informaticii să pună sub semnul întrebării evoluția și performanțele cu care informatica ne-a uimit și în același timp ne-a alintat, pînă în prezent ? Un răspuns probabil la această întrebare, sperăm să-l desprindeți dumneavoastră din aceste cîteva rînduri.

O undă de optimism străbate din cînd în cînd ; apariția unor arme a provocat întotdeauna și perfectarea unor mijloace de contracarare a efectului lor. O lege implacabilă, a evoluției, pare să guverneze și acest domeniu. Așadar, nu trebuie să ne temem ; și mai ales nu trebuie să cedăm în fața panicii create : mintea omenească are nevoie să gîndească în liniște.

Dincolo de ce trebuie și ce nu trebuie să faci, de ce este și ce nu este încă cunoscut despre virușii informatici există un proces de cercetare mult mai profund. S-ar putea ca tot biologicul să ne ofere modele. Direcții de acțiune există ; ele au început să se contureze. Lupta între viruși și contraviruși continuă. Contează numai pe ce plan și în care scopuri este dusă.

Lupta pe plan local, ca o funcție atribuită prin descentralizarea fecărei celule, pare să aibă șanse de izbîndă ceva mai ridicate. Deși, ar presupune creșterea dimensiunii fiecărui program cu o lungime considerabilă a codului executabil, pînă la urmă va fi acceptată această variantă în care fiecare program dispune de capacitatea de a se cunoaște, în linii mari, pe sine, de a distinge între propriu și străin.

Orice tentativă de grefare declanșată de o secvență virală ar putea fi atunci sesizată, cel puțin în faza rulării programului, cînd el devine pentru o cuantă de timp „viu". O imagine în miniatură a propriei structuri cheie ar trebui în acest caz să intre în zestrea fiecărui program executabil. Ar fi posibil astfel implementarea unui mecanism de tip imunitar, pentru contracararea oricărei încercări de manipulare a sa din exterior, prin infiltrarea unor secvențe parazite, agresive.

Problema virușilor este foarte diferită atît din punct de vedere operațional, cît și din punct de vedere tehnic. Dar poate cea mai importantă latură a acesteia este cea socială și educațională. Oamenii trebuie să fie răspunzători de acțiunile programelor pe care le elaborează.

Virușii indiferent de cauza care îi generează și de scopul pentru care au fost creați, rămîn o gravă manifestare antisocială, un „teribilism informatic" dăunător într-o lume din ce în ce mai dependentă de calculatoare.

BIBLIOGRAFIE SELECTIVĂ

1. *ANGHELOIU, I.; PATRICIU, V.; GYÖRFI, E., Securitatea și protecția informației în sistemele electronice de calcul*, Editura Militară, București, 1986.
2. *BURGER, J., VIRUS — la maladie des ordinateurs... De la protection du matérial á la protection juridique*, Micro Application, 1988.
3. *NORTON, P., Inside the IBM PC*, A Brady Boob, New York, 1986.
4. *ROSCH, W., Guide P.S.I. Material PC, PS et compatibles. Conception et fonctionnement*, Editions P.S.I., Gentilly, 1990.
5. *SHAPIRA, E.; SHERMAN, Y., Turbo Anti-Virus, V6.93A*, Carmel Software Engineering, 1990.
6. *TOWNSEND, C., Advanced MS-DOS. Expert Techniques for Programmers*, Haward W. Sams & Company, Indianapolis, 1989.
7. * * * *If*, nr. 1, oct., 1990.
8. * * * *Microsoft MS-DOS, User's Guide and User's Reference Operating System, Version 4.0*, Microsoft Corporation, 1988.
9. * * * *PC-Magazin*, nr. 1-4, 1990.
10. * * * *Soft & Micro*, nr. 68, nov. 1990.
11. * * * *Știință și tehnică*, nr. 4,8-9,10/1990.

Bun de tipar 10-06-1991.
Apărut 1991.

Tiparul executat sub comanda nr. 710
la I. P. „Filaret", str. Fabrica de chibrituri
nr. 9—11, București
România